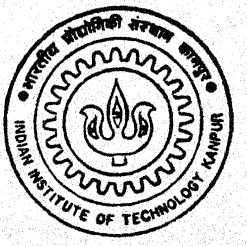
OPTIMAL DESIGN OF B - SPLINE CURVE AND SURFACE

by

Aniruddha A. Ambekar



DEPARTMENT OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

March, 1996

OPTIMAL DESIGN OF B-SPLINE CURVE AND SURFACE

A Thesis Submitted
in Partial Fulfilment of the Requirements
for the Degree of
MASTER OF TECHNOLOGY

by

Aniruddha A. Ambekar

to the

DEPARTMENT OF MECHANICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR

March, 1996

ME-1996-M-AMB-OPT

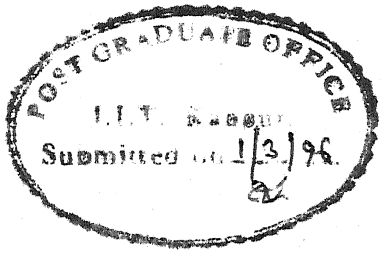
CERTIFICATE

It is certified that the work contained in the thesis entitled " **OPTIMAL DESIGN OF B-SPLINE CURVE AND SURFACE** " by *Aniruddha A . Ambekar* has been carried out under my supervision and it has not been submitted elsewhere for a degree.

Dhande
27-2-96

Dr. S. G. Dhande
Professor
Department of ME & CSE
IIT Kanpur

February, 1996



Dedicated
to
my Parents

Acknowledgements

With deep sense of gratitude, I take this opportunity to express my indebtedness and sincere thanks to Dr. Sanjay G. Dhande, who introduced me to the field of CAD. I thank him again for his invaluable guidance, constant encouragement and for providing a highly informal work-culture at CAD laboratory.

I am thankful to my ex-colleague P V M Rao for many useful discussions I had with him during the course of this work. I extend my thanks to Dr. K. Deb also for his help on GA part. I am also thankful to all my colleagues at CAD laboratory and the staff for their help and cooperation.

During my stay at I I T Kanpur, I have shared indelible memories of fun, excitement and affection with my friends and inmates of Hall iv ; V.Raghavan, Amarendra, Uday, Sanjeev, R. P. Gupta, Tushar, Shashikant, N.C.S. Reddy are to name a few. I wish them success in their future endeavor.

Lastly, I have deepest appreciation for the love and affection bestowed on me by my parents and my younger brother. They have always been a source of motivation for me.

Feb. , 1996 A. Ambekar

Table of Contents

List of Figures

List of Tables

Nomenclature

α	Rotation angle about X axis
β	Rotation angle about Y axis
B_i	Defining polygon vertex (for curves)
$B_{i,j}$	Defining polygon vertex (for surfaces)
[B]	Matrix of defining polygon vertices
[C]	Matrix of blending functions
Cp^1	Coordinate frame of prototype surface (Initial)
Cp^2	Coordinate frame of prototype surface (converged)
C^d	Coordinate frame of design surface
e_{rms}	RMS error
e_{total}	Total error
γ	Rotation angle about Z axis
k	Order of the curve
ku	Order of surface in u parametric direction
kv	Order of surface in v parametric direction
l	Translation along X axis
$M_{j,k}$	Basis functions in v parametric direction
m	Translation along Y axis
n	Translation along Z axis
$N_{i,k}$	Basis functions in u parametric direction

nu	Number of data points in u parametric direction
nv	Number of data points in v parametric direction
P(u)	Position vector of a point at parameter u
P_i	Point under consideration
[P]	Matrix of data points (curves)
Q(u,v)	Position vector of a point at parameter u and v
[Q]	Matrix of data points (surfaces)
$[Q^p]$	Matrix of data points on prototype surface
$[T_{Rx}]$	Rotation matrix about X axis
$[T_{Ry}]$	Rotation matrix about Y axis
$[T_{Rz}]$	Rotation matrix about Z axis
$[T_{Tr}]$	General translation matrix
[T]	Concatenated general transformation matrix
u_{li}	Generic u parameter corresponding to l^{th} population (curve)
u_{lij}	Generic u parameter corresponding to l^{th} population (surface)
v_{lij}	Generic v parameter corresponding to l^{th} population (surface)
X_i	Element of knot vector in u parametric direction
Y_i	Element of knot vector in v parametric direction

Abstract

Reckoff has defined ***'Reverse Engineering'*** as *"the act of creating a set of specifications for a piece of hardware by some one other than the original designer, primarily based upon analyzing and dimensioning a specimen or collection of specimens"*. This broad definition is based on the systems approach to design and encompasses a number of potential methodologies whereby an existing product is analyzed, prior to, or during, the development of a new product. The accurate physical replication of free form curves and sculptured surfaces is required for a variety of important industrial, consumer and medical applications. Often, the only available representation of a desired form or shape is a previously manufactured product or a naturally existing entity. The shape modelling, optimization and accurate reproduction of these forms is the subject of the present work. The objective here is two fold :

- To develop and to implement the methodology for free-form B-spline curve and surface fitting using the ***CMM*** scanned data and the subsequent optimization of the curves and the surfaces to improve the fit. The robust search technique ***'Real coded GA'*** with ***SBX*** as the cross over strategy has been used for optimization with simple bounds imposed on the variables.
- The best fit surface obtained in the previous stage is treated as the ***'design surface'*** for all the succeeding prototyping processes. The next part deals with the surface localization and the error evaluation of the manufactured surface with respect to the design surface. The problem has been formulated as an unconstrained nonlinear optimization problem. Nelder and Mead's ***'Simplex search method'*** is used as the search technique.

The implementation results are included to illustrate the approach. The graphical interface is provided at each stage using the **'Script file'** concept of **AutoCAD**.

Chapter 1

Introduction

1.1 Overview of Reverse Engineering

Global competitiveness is a key initiative for many companies striving to succeed in today's world-wide manufacturing society. In order to remain competitive many companies have initiated extensive programs to improve quality and to reduce the time-to-market. Most prominent among these programs are concurrent design, design reuse, rapid prototyping, and reverse engineering. Reverse engineering is a beneficial and a time reducing tool for the design of complex surfaces, particularly in the case of hand crafted or sculptured surfaces [1].

Sculptured surfaces are being integrated more and more in to the design of new products for many reasons including ergonomic optimization, aerodynamic advantage,and improved aesthetics. Optimized designs which maximize physical properties such as strength or heat resistance using the minimum amount of material often necessitate thin skin structure of complex form.

Many CAD systems now support the creation and use of free-form surfaces, and recent advances in surface and solid modelers have made widespread adoption of mathematically defined surfaces a reality. However, creating complex free-form surfaces from scratch utilizing solely these CAD tools to attain the desired geometry, topography and functional attributes is difficult and time consuming, requiring considerable skill, and usually taking several iterations involved with the building and testing of physical prototypes.

In engineering design, the use of existing forms such as standard off-the-shelf components and stock items, is always desirable. This is traditionally a good design practice and is particularly necessary today when the shortened product cycles dominate the design procedure [9].

In many applications, part, or all, of the desired surface form already exists physically as an existing manufactured component or naturally existing entity. Reverse engineering enables us to reuse existing forms, to rapidly capture the mathematical representation of that form, to reproduce it for reuse in a new product, and to reduce the time of product development cycle associated with products of complex surface form.

1.2 Applications of Reverse Engineering

The applications of reverse engineering for complex sculptured surfaces vary greatly [1] but the important among them are listed below :

Complex Sculptured Surface Generation : Components with sculpted or free-form surfaces have traditionally been sculpted or molded by skilled craftsmen. Similarly engineers often fashion manifolds and ducts until proper flow characteristics are achieved. In addition,a new design may be developed entirely from the prototype model.

Part Conversion : Many companies in order to remain competitive are implementing or upgrading their older, less capable CAD systems to more powerful 3D modelers. These companies are looking to convert old designs in to a usable 3D CAD format. Often, the only available representation that the company has of the current part is a 2D hand drawing or the NC machine code that was originally used to manufacture it.

Bench-marking : Most companies in competing industries carry out benchmark comparisons of competitor's products. Reverse engineering allows the engineers to input the facsimile of the competitor's product for carrying out in depth analysis.

Medical Applications : A reverse engineering approach is readily applied in applications where the complex and highly unique human surface must be created in order to obtain an exact fit for a joint or surgical implant design.

1.3 Problem Statement

The framework of a reverse engineering system has been illustrated in Figure 1.1 . The objective here is to develop a physical prototype starting from a existing manufactured component. The activities can be divided in to following stages :

1. Scanning : It involves scanning the existing manufactured object with the help of laser scanner or the coordinate measuring machine (CMM). The data generated is point data consisting of

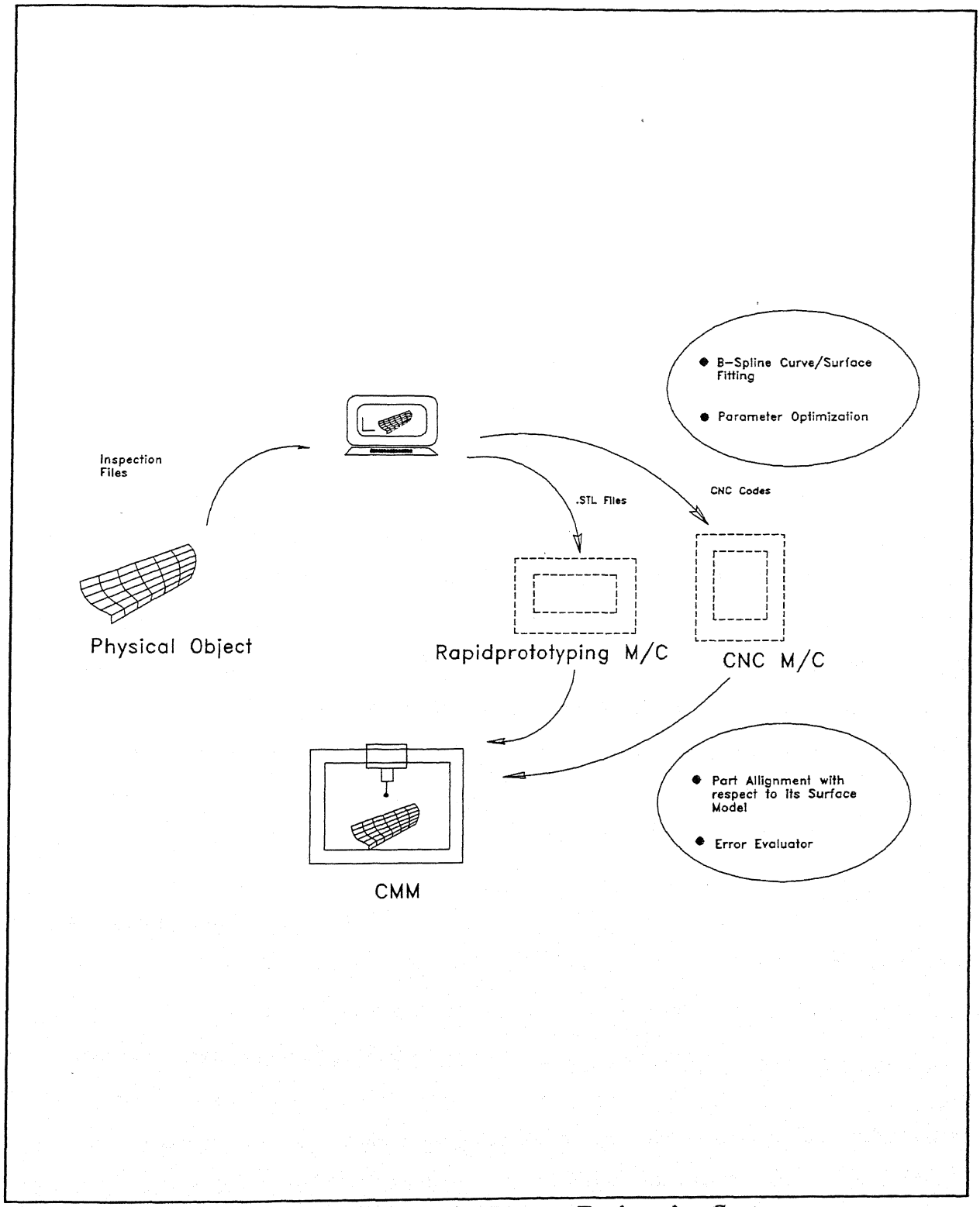


Figure 1.1 Framework of a Reverse Engineering System

x,y,z values of measured points.

2. Fitting : The measured point data is to be converted into mathematically defined curves and surfaces. The surface model generated at this stage then serves as *'design surface'* for rest of the prototyping activities.

3. Postprocessing : The design surface generated in previous stage is used to generate either the NC codes or the .STL files depending on the prototyping machine. Latter is used in case of a rapid prototyping machine.

4. Surface localization : This activity is needed in order to ascertain the proximity of the prototype surface with respect to the design surface. The feedback of the error at this stage may be used to undertake some corrective measures, if desired.

The domain of study of the present work is second and the fourth activity, namely Fitting and the Surface Localization.

1.3.1 B-Spline Curve and Surface Fitting

B-Splines being the de-facto industry standard for most of the commercially available CAD packages, they are the most obvious candidate for fitting. Unfortunately this representation is not very amenable for curve and surface fitting, as basically, it relies on the guiding polygon information to be supplied by the user. As a part of this thesis work a GA based approach has been developed and implemented for curve and surface fitting using B-Splines, starting from digitized point data. Open freeform curves and surfaces have been considered for fitting. Real coded GA with SBX as the crossover strategy have been used for implementation.

1.3.2 Surface Localization

The success of reverse engineering activity depends on the proximity of prototype surface with the design surface. The error between the two surfaces can arise because of

1. Positioning uncertainty while measuring
2. Manufacturing defects

The above problem has been formulated as an unconstrained nonlinear optimization problem with the explicit error expression as the objective function and Nelder and Mead's Simplex search procedure as the search technique. The solution is essentially the true location

of the prototype surface with respect to the design surface and the magnitude of the form error.

1.4 Literature Survey

Among the previous studies on curve and surface fitting the most applicable to the current investigation are those by Rogers & Fog [2], Hoschek [5], Sarkar & menq [4], Duffie & Feng [11], Green & Philpott [1], Ma & Kruth [3].

Rogers & Fog [2] suggested a least square fitting approach of curve and surface fitting for B-Splines. The improvement in the fit is achieved by iteratively improving the parameter values using a first order Taylor correction for the error expression. Hoschek [5] proposes an iterative fitting scheme for the B-Splines where the aim is to make the error vectors normal to the fitted surface, and thereby improving the fit. Sarkar & Menq [4] have proposed Levenberg-Marquardt optimization scheme for B-Spline surface fitting with variable bounds .

Duffie & Feng [11] use hermite patches for surface fitting while iterating on the coefficient matrix to improve the fit. Green & Philpott [1] use a similar scheme. Ma & Kruth have used a base curve/base surface approach where the measured points are projected on the base curve or base surface to obtain approximate parameter values. These parameter values are modified depending on the error vector. The base curve and the surface must follow certain properties such as unique local mapping, smoothness and closeness.

The iterative schemes of Rogers & Fog [2] and Hoschek [5] are simple to implement but slower in convergence. Menq & Sarkar [4] suggest an optimization scheme which is faster in convergence but relies on numerically computed Jacobians. Hermite patch approach [1,11] requires the knowledge of derivatives and cross derivatives.

All the above mentioned schemes, except that of Ma & Kruth [3], suffer from the drawback that they start under an assumption of uniform knot vector ignoring the distribution of the measured points. This may result in a badly scaled, and sometimes even a singular matrix of blending functions. Ma & Kruth [3] have used an average knot scheme which is primarily based on the distribution of the measured points. Their method, works better than previous methods. But the disadvantage of their approach is that, their method requires the base curve/base surface to be specified by the user, which is not very intuitive on the part of user. Further the convergence depends on how close the base surface is to the surface to be fitted.

In the present study the average knot approach has been retained but the knot vector is adaptively selected. The robust search scheme Real coded GA with SBX crossover is used for the fitting. A novel initialization scheme is implemented for the parameterization.

Gunnarsson & Prinz [6] have studied the part localization problem in the context of manufacturing applications. Pahk, Kim, Hong, and Kim [7] suggest an integrated approach for computer aided inspection for mold manufacturing. A similar formulation is found to be applicable in reverse engineering application. The problem has been formulated as an unconstrained nonlinear optimization problem and is solved by Simplex search method .

1.5 Organization of the Thesis

The current chapter is the introduction to the thesis work. Chapter 2 deals with the basic mathematical formulation for curve/surface fitting and discusses the optimization method ' Real Coded Genetic Algorithm ' in the context of its application. Chapters 3, 4 and the 5 mainly constitute the body of the thesis. Chapter 3 and 4 cover the implementation details and the examples related to curve/surface fitting problem. Mathematical formulation of surface localization problem has been dealt in chapter 5 itself along with the implementation details and examples. Sixth chapter concludes the thesis work.

Chapter 2

Mathematical Formulation : Curve/Surface Fitting and Optimization

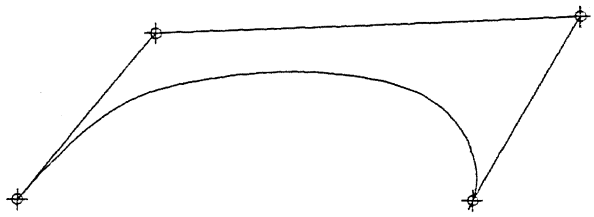
2.1 Curve/Surface Design with B-splines

Owing to the advantages in terms of flexibility and control, B-splines have become today the industry standard for CAD database representation. In the conventional design practice the B-splines are used mostly for '*approximating*' the data, i.e., given a set of data points, a B-spline curve/surface can be found which is reasonably close to the data and enjoys certain properties such as order and the continuity, as desired by the designer. The data point information supplied by designer is in the form of defining polygon vertices. The B-spline curve/surfaces have been derived from B-spline basis. This basis is generally nonglobal [13]. The nonglobal behavior of the B-spline basis is due to the fact that each guiding polygon vertex B_i is associated with a unique basis function. Thus each vertex affects the shape of the curve/surface only over a range of parameter values where the associated basis function is non-zero. The B-spline basis also allows the order of the resulting curve/surface to be changed without changing the number of the defining polygon vertices.

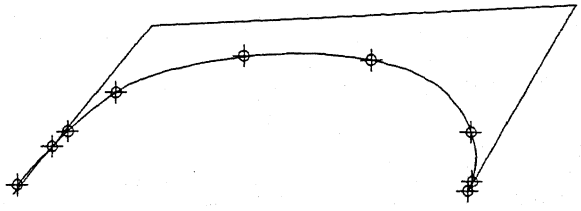
In reverse engineering applications the success of the modeling depends on how closely the curve or the surface conforms to the data points, i.e., this is an '*interpolation*' problem as opposed to the '*approximation*' problem discussed above. Figure 2.1 illustrates this difference. In the following section, the difference between approximation and interpolation has been outlined under the heading of Forward Design and the Inverse Design respectively.

2.1.1 Forward Design

a) Curve :



a. Forward Design (Polygon Points Specified)



b. Inverse Design (Curve Points Specified)

Figure 2.1 Forward and Inverse Design Problems

Let $P(u)$ be the position vectors of a generic point on the curve as a function of the parameter u. A B-spline curve is given by

$$P(u) = \sum_{i=1}^{n+1} B_i N_{i,k}(u) \qquad u_{min} \le u \le u_{max}, \quad 2 \le k \le n+1 \tag{2.1}$$

where the B_i are the position vectors of the $n+1$ defining polygon vertices and the $N_{i,k}$ are the normalized B-spline basis functions [14].

For the i^{th} normalized B-spline function of order k (degree $k-1$), the basis functions are defined by the Cox-deBoor recursion formulas.

Specifically,

$$N_{i,1}(u) = \begin{cases} 1 & \text{if } x_i \le u < x_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

and

$$N_{i,k}(u) = \frac{(u - x_i)N_{i,k-1}(u)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - u)N_{i+1,k-1}(u)}{x_{i+k} - x_{i+1}} \tag{2.3}$$

The values of x_i are elements of a knot vector satisfying the relation $x_i <= x_{i+1}$. The parameter u varies from u_{min} to u_{max} along the curve $P(u)$. The convention $0 / 0 = 0$ is adopted.

Formally a B-spline curve is defined as a polynomial spline function of order k (degree $k-1$) as it satisfies the following two conditions :

1. The function $P(u)$ is a polynomial of degree $(k-1)$ over the interval $x_i <= u < x_{i+1}$.
2. $P(u)$ and its derivatives of order 1,2, ... , $k-2$ are all continuous over the entire curve.

Thus, for example, a fourth order B-spline is a piecewise cubic curve.

The B-spline functions have the following properties [15] :

$$
\begin{aligned}
&\textit{Partition of unity} : \quad \sum_{i=1}^{n+1} N_{i,k}(u) = 1 \\
&\textit{Positivity} : \quad N_{i,k}(u) \geq 0 \\
&\textit{Local support} : \quad N_{i,k}(u) = 0 \quad \textit{if } u \notin [u_i , u_{i+k+1}] \\
&\textit{Continuity} : \quad N_{i,k}(u) \textit{ is } (k-2) \textit{ times continuously differentiable}
\end{aligned}
\tag{2.4}
$$

The first property ensures that the relationship between the curve and its defining control points is invariant under affine transformation. The second property guarantees that the curve segment lies completely within the convex hull of B_i. The third property indicates that each segment of B-spline curve is influenced by only k control points or each control point affects only k curve segments.

Equations 2.2 and 2.4 clearly show that the choice of knot vector has a significant influence on the resulting B-spline curve. The only requirement for a knot vector is that it satisfies the relation $x_i <= x_{i+1}$; i.e., it is a monotonically increasing series of real numbers. Fundamentally three types of knot vectors are used : uniform, open uniform (or open) and nonuniform.

In a uniform knot vector, individual knot values are evenly spaced. Examples are

[0 1 2 3 4]

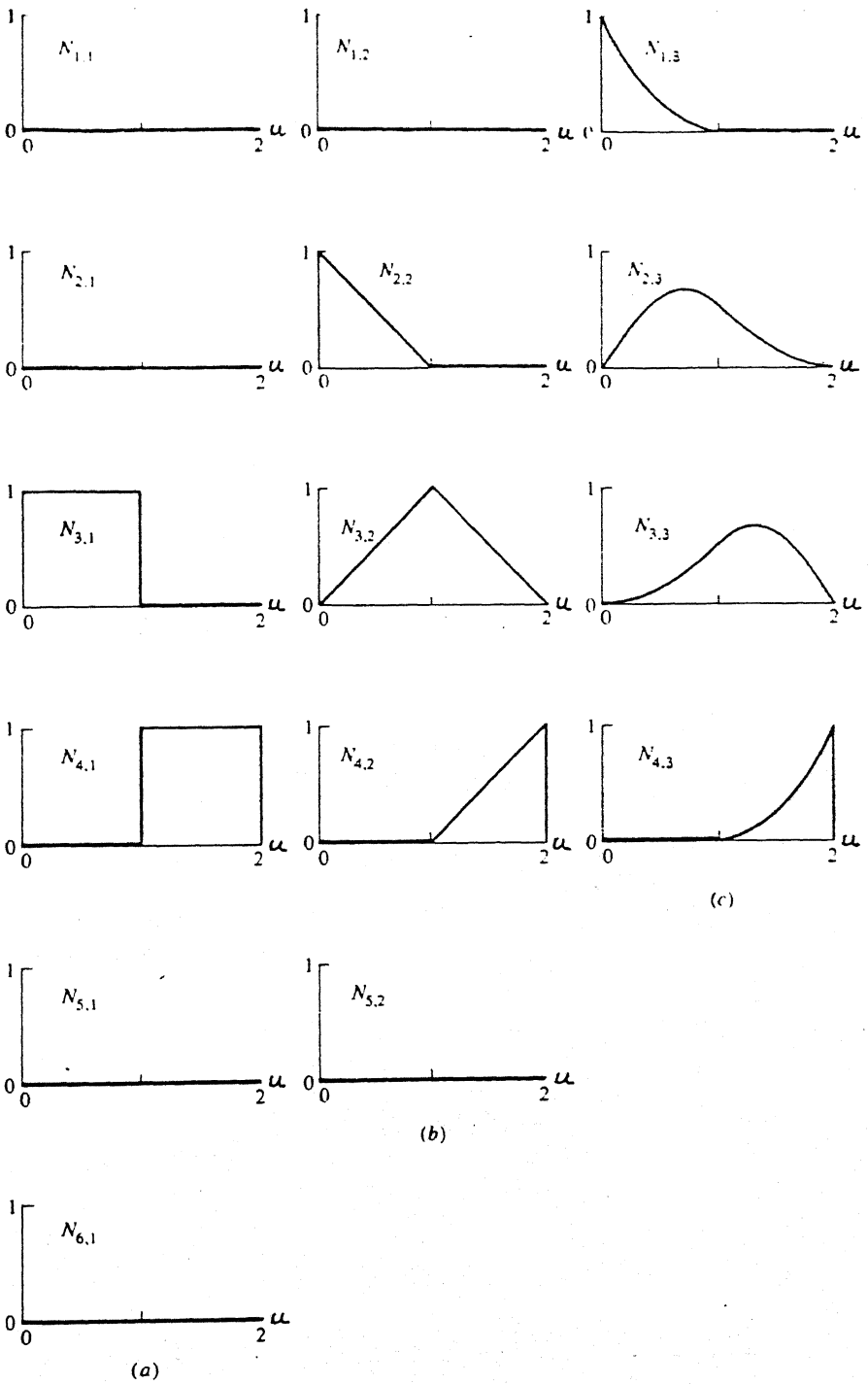
An open uniform knot vector has multiplicity of knot values at the ends equal to the order k of the B-spline basis function. Internal knot values are evenly spaced. An example is

[0 0 1 2 2] (for $k=2$)

Formally, an open uniform knot vector is given by

$$
\begin{aligned}
x_i &= 0 \qquad 1 \leq i \leq k \\
x_i &= i-k \qquad k+1 \leq i \leq n+1 \\
x_i &= n-k+2 \qquad n+2 \leq i \leq n+k+1
\end{aligned}
\tag{2.5}
$$

The basis function buildup for open uniform case has been shown in Figure 2.2. Finally nonuniform knot vectors may have either unequally spaced and/or multiple internal knot values. They may be periodic or open. The examples are



n+1= 4; (a). k=1; (b.) k=2; (c.) k=3

Figure 2.2 Basis Function Buildup (Open Uniform Case)

[0 0 0 1 1 2 2 2]

[0 1 2 2 3 4]

[0 0.28 0.5 0.72 1]

In the present implementation, open nonuniform knot vectors have been used. Figure 2.3 illustrates the above three types of knot vectors.

b) Surface :

The Cartesian product B-spline surface is defined by

$$Q(u,v) = \sum_{i=1}^{n+1} \sum_{j=1}^{m+1} B_{i,j} N_{i,k}(u) M_{j,l}(v) \tag{2.6}$$

where $N_{i,k}(u)$ and $M_{j,l}(v)$ are the B-spline basis functions in biparametric u and v directions [14].The definitions of the basis functions given earlier is repeated here for the sack of convenience.

$$N_{i,1}(u) = \begin{cases} 1 & \text{if } x_i \le u < x_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

$$N_{i,k}(u) = \frac{(u - x_i) N_{i,k-1}(u)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - u) N_{i+1,k-1}(u)}{x_{i+k} - x_{i+1}} \tag{2.8}$$

and

$$M_{j,1}(v) = \begin{cases} 1 & \text{if } y_j \le v < y_{j+1} \\ 0 & \text{otherwise} \end{cases} \tag{2.9}$$

$$M_{j,l}(v) = \frac{(v - y_j) N_{j,l-1}(v)}{y_{j+l-1} - y_j} + \frac{(y_{j+l} - v) M_{j+1,l-1}(v)}{y_{j+l} - y_{j+1}} \tag{2.10}$$

where the x_i and the y_j are elements of the knot vectors already discussed in the context of curve design. Since the B-spline basis is used to describe both the boundary curves and to blend the interior to the surface the properties of B-spline curves can be extended to the surfaces as well. The property of the invariance under affine transformation, convex hull property, hold good here also and the third property says, that the influence of each control polygon vertex is limited to $\pm k/2$ and $\pm l/2$ spans in both parametric directions respectively.

2.1.2 Inverse Design

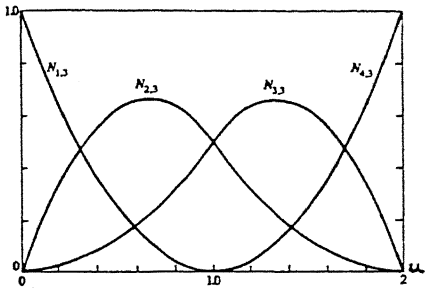
a) Curve :

The previous section discussed the generation of a B-spline curve from its defining polygon. Here, determining a polygon that generates a B-spline curve through a set of data points is considered. The problem is shown schematically in Figure 2.1 (b).

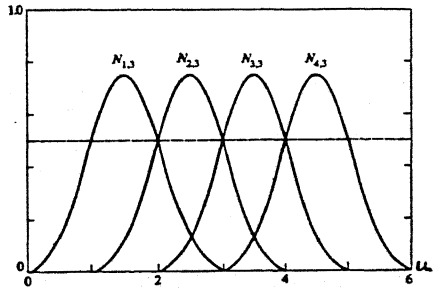
If a data point lies on the B-spline curve, then it must satisfy Equation 2.1. Writing Equation 2.1 for each of j data points yields

$$\begin{aligned} P_1(u_1) &= N_{1,k}(u_1)B_1 + N_{2,k}(u_1)B_2 + \cdots + N_{i,K}(u_1)B_i \\ P_2(u_2) &= N_{1,k}(u_2)B_1 + N_{2,k}(u_2)B_2 + \cdots + N_{i,K}(u_2)B_i \\ &\vdots \\ P_j(u_j) &= N_{1,k}(u_j)B_1 + N_{2,k}(u_j)B_2 + \cdots + N_{i,K}(u_j)B_i \end{aligned} \qquad (2.11)$$

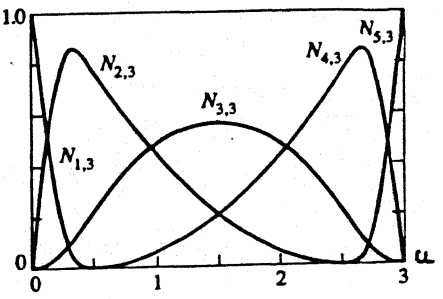
where $2<=k<=n+1 = j$. This set of equations is more compactly written in matrix form as



Open Uniform Basis Function
n+1=4; k=3; X=[0 0 0 1 2 2 2]



Periodic Uniform Basis Function
n+1=4 ;k=3; X=[0 1 2 3 4 5 6]



Open Non Uniform Basis Function
n+1=5 ;k=3;X=[0 0 0 .4 2.6 3 3 3]

Figure 2.3 Types of Basis Functions

$$[P] = [C][B] \tag{2.12}$$

$$[C] = \begin{bmatrix} N_{1,k}(u_1) & \cdots & \cdots & N_{i,k}(u_1) \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ N_{1,k}(u_j) & \cdots & \cdots & N_{i,k}(u_j) \end{bmatrix} \tag{2.13}$$

If $2 <= k <= n+1 = j$, then the matrix *[C]* is a square matrix and the defining polygon is obtained directly by the matrix inversion, i.e.,

$$[B] = [C]^{-1}[P] \qquad 2 \le k \le n+1 = j \tag{2.14}$$

In this case the resulting B-spline curve passes through each and every data point, i.e., a curve fit is obtained. Although the continuity of the resulting curve is everywhere C^{k-2} , it may not be ' smooth ' or ' fair '.

A fairer curve is obtained by specifying fewer defining polygon points than data points. i.e., $2 <= k <= n+1 < j$. Here, *[C]* is no longer square , the problem is over-specified and can be solved only in some mean sense. Recalling that a matrix times its transpose is always square, the defining polygon vertices for a B-spline curve that fairs or smooths the data is given by

$$\begin{aligned} [P] &= [C][B] \\ [C]^T[P] &= [C]^T[C][B] \\ [B] &= [[C]^T[C]]^{-1}[C]^T[P] \end{aligned} \tag{2.15}$$

Both of these techniques assume that the matrix *[C]* is known. Provided that the order of the B-spline basis, the number of defining polygon vertices $n+1$ and the parameter values are known matrix *[C]* can be obtained.

To find out the parameterization is not a simple task if the fit has to be an accurate one. One useful parameterization model is based on the chord length approximation.

$$u_1 = 0$$

$$\frac{u_l}{u_{max}} = \frac{\sum_{s=2}^{l} |P_s - P_{s-1}|}{\sum_{s=2}^{j} |P_s - P_{s-1}|} \qquad l \geq 2 \tag{2.16}$$

Here j is the total number of data points.The maximum parameter value u_{max} is generally taken as the maximum value of the knot vector.

b) Surface :

Section 2.1.1 (b) discusses the generation of B-spline surfaces from a known defining polygon net. Inverse problem is also of importance; i.e., given a known set of data points on the surface, determine the best polygon net which interpolates that data [14].

Here $Q(u,v)$ are the known surface data points; the $N_{i,k}(u)$ and $M_{j,l}(v)$ basis functions can be determined for a known order and a known number of defining polygon net vertices in each parameter direction provided that the parameter values u, v are known at the surface data points. Writing out Equation (2.6) for a single surface data point yields

$$Q_{1,1}(u_1,v_1) = N_{1,k}(u_1)[M_{1,l}(v_1)B_{1,1} + M_{2,l}(v_1)B_{1,2} + \cdots + M_{m+1}(v_1)B_{1,m+1}] +$$

$$\vdots$$

$$N_{n+1,k}(u_1)[M_{1,l}(v_1)B_{n+1,1} + M_{2,l}(v_1)B_{n+1,2} + \cdots + M_{m+1}(v_1)B_{n+1,m+1}] \tag{2.17}$$

where for an $r \times s$ topologically rectangular set of data $2 <= k <= n+1 < r$ and $2 <= l <= m+1 < s$. Writing the equation in matrix form, we get

$$[Q] = [C][B] \tag{2.18}$$

where $C_{i,j} = N_{i,k}M_{j,l}$. For $r \times s$ topologically rectangular surface point data, $[Q]$ is an $r*s \times 3$ matrix containing the 3D coordinates of the surface point data. $[C]$ is an $r*s \times n*m$ matrix of the B-spline basis functions, and $[B]$ is an $n*m \times 3$ matrix of the 3D coordinates of the required polygon net points.

If $[C]$ is square, the defining polygon net is obtained directly by inversion

$$[B] = [C]^{-1}[Q] \tag{2.19}$$

In this case the resulting surface passes through each data point. Although the resulting surface will be everywhere C^{k-2}, C^{l-2} continuous, it may not be fair. In general, fewer the polygon points the fairer the surface.

If $[C]$ is not square the problem is over-specified and the solution can only be found in some mean sense.

$$[B] = [[C]^T[C]]^{-1}[C]^T[Q] \tag{2.20}$$

The u and v values at each surface points have to be found out by using some parameterization technique. One useful technique is chord length parameterization.

For u parametric direction

$$\begin{aligned} u_1 &= 0 \\ \frac{u_l}{u_{\max}} &= \frac{\sum_{g=2}^{l} |P_{g,s} - P_{g-1,s}|}{\sum_{g=2}^{r} |P_{g,s} - P_{g-1,s}|} \end{aligned} \tag{2.21}$$

For v parametric direction

$$v_1 = 0$$

$$\frac{v_l}{v_{\max}} = \frac{\sum_{g=2}^{l} |P_{r,g} - P_{r,g-1}|}{\sum_{g=2}^{s} |P_{r,g} - P_{r,g-1}|} \qquad (2.22)$$

where u_{max} and v_{max} are the maximum values of the appropriate knot vectors.

2.2 Need of Parameter Optimization

The B-spline interpolation problem is usually stated as " given the data points P_i and the parameter values u_i , find out the curve/surface passing through these points ". This is the mathematicians way of describing a problem . In practice, the parameter values are rarely given and must be made up somehow. The easiest way to determine the u_i is simply to set

$$u_i = \frac{i-1}{s-1} \qquad 1 \le i \le s \qquad (2.23)$$

This is called uniform parameterization. Above equation holds good for ' s ' curve points. This method is too simplistic to cope with most of the practical situations. The reason for overall poor performance of the uniform parameterization can be blamed on the fact that it ignores the ' geometry ' of the data points [16]. The following is the heuristic explanation of the fact.

We can interpret the parameter ' u ' of the curve as time ; as the time passes from u_0 to u_l the point $P(u)$ traces out the curve from $P(u_0)$ to $P(u_l)$. With uniform parameterization $P(u)$

spends the same amount of time between any two data points, irrespective of their relative distances. Going by the above analogy, the method is bound to fail when the spacing between the points is uneven.

The chord length parameterization is slightly better than the uniform parameterization as it takes in to account the relative distance between the data points.

$$u_i = u_{i-1} + \frac{|P_{j+1} - P_j|}{\sum_{j=1}^{s-1} |P_{j+1} - P_j|} \qquad 2 \le i \le s \tag{2.24}$$

Another parameterization has been named as ' centripetal ' by E. Lee [16] . If we recall the analogy presented for uniform parameterization , here the resulting motion of a point on the curve tends to ' smooth out ' the variations in the centripetal force acting on it.

$$u_i = u_{i-1} + \frac{|P_{j+1} - P_j|^{\frac{1}{2}}}{\sum_{j=1}^{s-1} |P_{j+1} - P_j|^{\frac{1}{2}}} \qquad 2 \le i \le s \tag{2.25}$$

There is probably no best parameterization , since any of the above method can be defeated by suitably choosen data points. Hence there is a need to evolve optimum set of parameters which in turn will generate curves/surfaces interpolating the given data points closely.

2.3 Real Coded Genetic Algorithms

Genetic algorithms (GAs) are nontraditional search and optimization techniques based on an analogy with natural genetics [18]. They have been applied successfully in to a wide variety of problems, and have yielded results which were either not possible or were very difficult to achieve using traditional methods. There are certain salient features of GAs which distinguish them from traditional methods.

1. GAs have *global perspective*, i.e., they dont get easily stuck in to local optima.
2. GAs are *robust* techniques i.e., they can be applied to many classes of optimization problems.

3. GAs use a *population* of solution points, and not a single point.
4. GAs use *probabilistic* rules for refinement of solution points and not the *deterministic* rules.
5. GAs dont require *gradient* or any other auxiliary information about the problem.
6. Simple GAs usually work with a coding of variables and not the variables themselves. But It has been well established that this feature is not absolutely necessary. Infact, `Real coded genetic algorithms' work directly with the variables.

Real coded GAs offer certain advantages over their binary counterparts (Simple GAs), when the search domain is continuous. Problem of fixed precision level, which is a shortcoming of binary coded GAs is not applicable in case of real coded GAs. Search power of real coded GAs is found to better than binary GAs in the continuous search domain. A newly developed SBX crossover has been found to enhance the flexibility and the search power of the real coded GAs [19].

2.3.1 Working of Real Coded GAs

A pseudo code for GAs is presented below

```
initialize( );
evaluation( );
while (termination criteria are unsatisfied)
{
        reproduction( );
        crossover( );
        mutation( );
        evaluation( );
}
```

In GA terminology a set of design variables exist as a concatenated string. The objective function value for such a string is called its *fitness*. First, a population of randomly generated strings is created. The number of strings in a population is called population size and is usually supplied by the user. For more complex problems, larger population sizes are normally used. Once the population is created each string in the population is evaluated and is assigned its

fitness value, which is a measure of its *goodness*. For a maximization problem the objective function value itself is treated as the fitness value. After this the three main operators – reproduction, crossover, and mutation , are applied on the population and it is again evaluated. This process is continued till termination criteria are satisfied. Each cycle of these three operators constitutes a *generation*. The three operators are described below

- **Reproduction** : This operator selects better strings from the population and sends copies of these strings in a temporary population called *mating pool*. This is analogous to Darwin's *survival of the fittest* principle. Usually a probabilistic approach is used to select the strings. There are a number of schemes which can be used for this purpose, e.g., proportionate selection, tournament selection and the stochastic remainder selection.
- **Crossover** : The reproduction operator can't generate new solution points. The crosssover operator is analogous to recombination of genes of parents in biological system. The main purpose of the crossover operator is to search the parameter space. Other aspect is that the search needs to be performed in such a way that the information stored in the parent string is maximally preserved. Many crossovers have been designed for real coded GAs, but for the present implementation SBX crossover [19] is used which is a simulated version of binary crossover. Details of the SBX crossover can be found in Appendix C.
- **Mutation** : It is considered to be a secondary operator in GAs, and its main function is to guard the population against the premature convergence. This operator is applied on a single parent point (as in traditional GAs) and results in a perturbance of its parameter values. This perturbance is calculated using a probability distribution which has the probability such that small perturbations are more probable than the large ones. This operator is exercised with a very small probability, and it is primarily meant for keeping diversity in population.

GAs are increasingly being used to solve geometric problems because of their ability to handle complex objective functions, global perspective and simple mechanism of working. In fitting problem inparticular, the objective function is highly nonlinear . In addition, there are many inherent constraints in the B-spline representation. For example, the blending functions are defined only over a range of parameter values. This fact coupled with the large number of

variables make this problem very difficult for the conventional gradient based methods.

Real GAs perform better than the simple GAs when the search space is continuous. Chapter 3 and 4 discuss the GA based B-spline curve and surface fitting respectively.

Chapter 3

GA Based Method for Curve Fitting

This chapter discusses the GA based method for B-Spline curve fitting. The input is a set of measured data points, each specified by x,y,z value. The objective is to interpolate a B-Spline curve through it such that the fitting error will be minimum. To evaluate the effectiveness of the method, a series of data points from a mathematically known curve are used as input. The problem has been formulated as an unconstrained nonlinear optimization problem with variable bounds . A novel initialization scheme has been presented for the set of parameter values and average knot scheme has been implemented for knot vector generation. The selection of the knot vector is made based on fitness ranking of the parameter values.

3.1 Parameterization Model

Parameterization is the process of assigning parameter values to the measured point data. Given an ordered set of points, P_i , the parameter values u_i can be found by any of the uniform, chord length or the centripetal parameterization discussed in Section 2.2 of previous chapter.

However the accuracy of the fit depends heavily on the type of parameterization model adopted.

A generalization of the above three models has been proposed by Lee [12] in his paper.

$$u_i = u_{i-1} + \frac{|P_i - P_{i-1}|^e}{\sum_{j=1}^{s-1} |P_{j+1} - P_j|^e} \qquad 2 \le i \le s \tag{3.1}$$

where s is the total number of data points specified on the curve. It is interesting to note that for $e = 0, .5, 1$; the above parameterization reduces to uniform, centripetal, chord length respectively.

A population initialization scheme has been formulated using above parameterization model. If *pop_size* is the number of populations and s is the number of specified data points on the curve then we can represent a generic population $u_{l,i}$ as

for(l=0 to popsize)

{

for (i=0 to s)

{

e = random number between 0 and 1

$$u_{l,i} = u_{l,i-1} + \frac{|P_{l,i} - P_{l,i-1}|^e}{\sum_{j=1}^{s-1} |P_{l,j+1} - P_{l,j}|^e} \qquad 2 \le l \le s \tag{3.2}$$

}

}

This population scheme offers following advantages.

- We can use a number of good points available to start the optimization routine. This reduces computational burden.
- The B-Spline basis functions are defined only if parameter values u_i follow certain properties such as, they should be monotonically increasing numbers between *0* and maximum value of the knot vector X_{n+k} . This method of initialization ensures that the infeasible points are never generated.

3.2 Objective Function Formulation

Let P_i be the measured data points, supplied by the user and P_i^c be the points on the interpolated B-Spline curve evaluated at the points corresponding to the parameter u_i .

Then ,

$$[P^c] = [C][[C]^T[C]]^{-1}[C]^T[P] \tag{3.3}$$

Error in x, y, z coordinates can be stated as

$$\delta x = |P_x^c - P_x|$$
$$\delta y = |P_y^c - P_y| \quad (3.4)$$
$$\delta z = |P_z^c - P_z|$$

Euclidean error expression at a particular point will be

$$e_i = \sqrt{\delta x^2 + \delta y^2 + \delta z^2} \quad (3.5)$$

Total error expression will be

$$e_{total} = \sum_{i=1}^{s} e_i \quad (3.6)$$

RMS error expression will be

$$e_{rms} = \sqrt{\sum_{i=1}^{s} \{\frac{e_i^2}{s}\}} \quad (3.7)$$

Optimization problem is formulated as " minimize RMS error where the design variables are parameter values u_i and the variables are bounded between 0 and the maximum value of the knot vector X_{n+k} ".

3.3 Knot Vector Selection

It should be noted that the parameterization of data points for fitting or interpolation and the parameterization of the knot values i.e., selection of $X = \{X_1, X_2, X_3, \ldots, X_{n+k}\}$ for B-Spline curve are closely related. This case is different from that of polynomial curve fitting where only one curve segment is considered . With a knot selection method consistent with the parameterization and the nature of the data points , one can use any kind of parameterization method to assign the parameter values to data points.

Average knot selection scheme has been used in present implementation. It generates open

nonuniform knot vector. This method offers some distinct advantages over uniform or chordal knot selection schemes.

- Usually while scanning an object measurement points are taken densely in the areas where the curvature is sharper. This method allocates more knots to places where the curve changes rapidly.
- With average knot method, it is almost sure that $[C]^T[C]$ matrix will not become singular or illconditioned. This condition is very important because during the optimization run $[C]^T[C]$ matrix is inverted in each function evaluation. Schoenberg - Whitney conditions for curve fitting are discussed in appendix A.

The knot selection is made depending on the fitness ranking of the population. Recalling that the population of u_i is generated using $e \in [0,1]$. The initial population is evaluated corresponding to $e = 0, .25, .5, 1$. The population having the best fitness is used to generate the knot vector using average knot scheme. Once the knot vector is selected it is retained throughout the execution of the optimization run.

Details of average knot selection scheme are provided in appendix B.

3.4 Implementation Details :

The algorithm of the method can be summarized as

Initialize population using the exponential initialization scheme;
Evaluate the fitness corresponding to e=0,.25,.5,1 to decide upon the population, to generate knot vector;
Find the knot vector using average knot vector scheme;

(while termination criteria are unsatisfied){
Reproduction ;
Crossover (using SBX strategy);
Mutation;
Evaluation;
}

3.5 Computational Aspects :

Software has been developed based on above formulation . First fit solution can be obtained by

B-Spline curve fitting technique (Section 2.1.2 (a)). Subsequent optimization is carried out using real coded GA with SBX crossover (Appendix C). The termination criteria is the reduction in error value or the maximum number of generations. The inversion of the blending function matrix is carried out using Cholesky decomposition method. The graphical output is created in the form of script files.

3.6 Examples and Discussion

Two examples are presented for the case of B-Spline curve fitting. The input data points are simulated in the sense that, they have been generated using a curve of known mathematical representation. The curves are fitted using user supplied information regarding the number of defining polygon vertices, the order of the curve, and the information needed for optimization run i.e., population size, the maximum number of generations, probability of crossover, probability of mutation and the distribution index. The uniform crossover on all the variables is implemented.

Example 1 Planar Curve: The first example of curve fitting involves sharp curvature changes as well as highly uneven spacing of measured data points (Figure 3.1 (a)). The number of design variables are equal to the number of parameter values i.e., 45 in this case. The first fit results of B-Spline curve clearly indicate the inadequacy of the simple inverse solution approach. The curve deviates from the data points at the points of sharp curvature variations where the spacing of the data points is very dense. Figure 3.1 (b) shows the results after the optimization run. The curve now conforms to the measured point data to a reasonable level of accuracy. Figure 3.2 shows the convergence pattern of the rms error with the number of generations. It is noteworthy that the rate of convergence decreases as the optimization proceeds and after the 200^{th} generation there is no reduction in error. Table 3.1 shows the input as well as the output data.

Example 2 Space Curve: The example again illustrates uneven spacing of data points. As the total number of data points are 47, the number of design variables in this problem are 47 . The first fit result deviates considerably from the data points (Figure 3.3 (a)). The result after the optimization run are fairly acceptable (Figure 3.3.(b)). Figure 3.4 , shows the convergence plot of the rms error with the number of generations. Here, the error becomes almost constant after

150th generation. Table 3.2 shows input and the output data.

Input		Output	
Number of Measured points	45	Initial Error	Final Error
Number of Polygon Points	16	.0598	.0049
Order	3		
Population Size	60		
Number of Generations	400		
Probability of Crossover	1		
Probability of Mutation	.01		
Distribution Index for SBX	2		

Table 3.1 Numerical data for example 1.

Input		Output	
Number of Measured points	47	Initial Error	Final Error
Number of Polygon Points	12	.215	.0478
Order	3		
Population Size	60		
Number of Generations	400		
Probability of Crossover	1		
Probability of Mutation	.01		
Distribution Index for SBX	2		

Table 3.2 Numerical Data for Example 2.

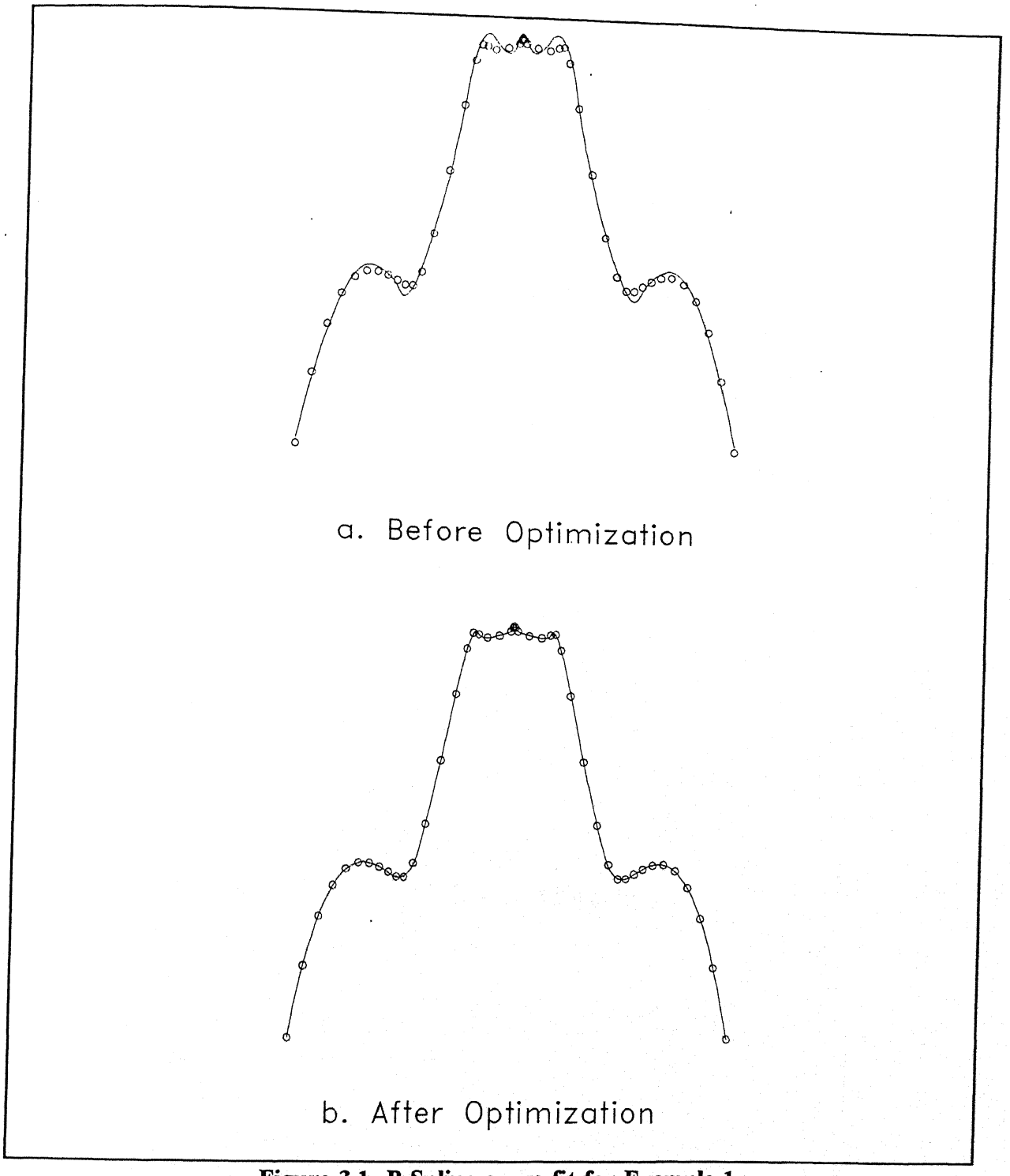


Figure 3.1 B-Spline curve fit for Example 1

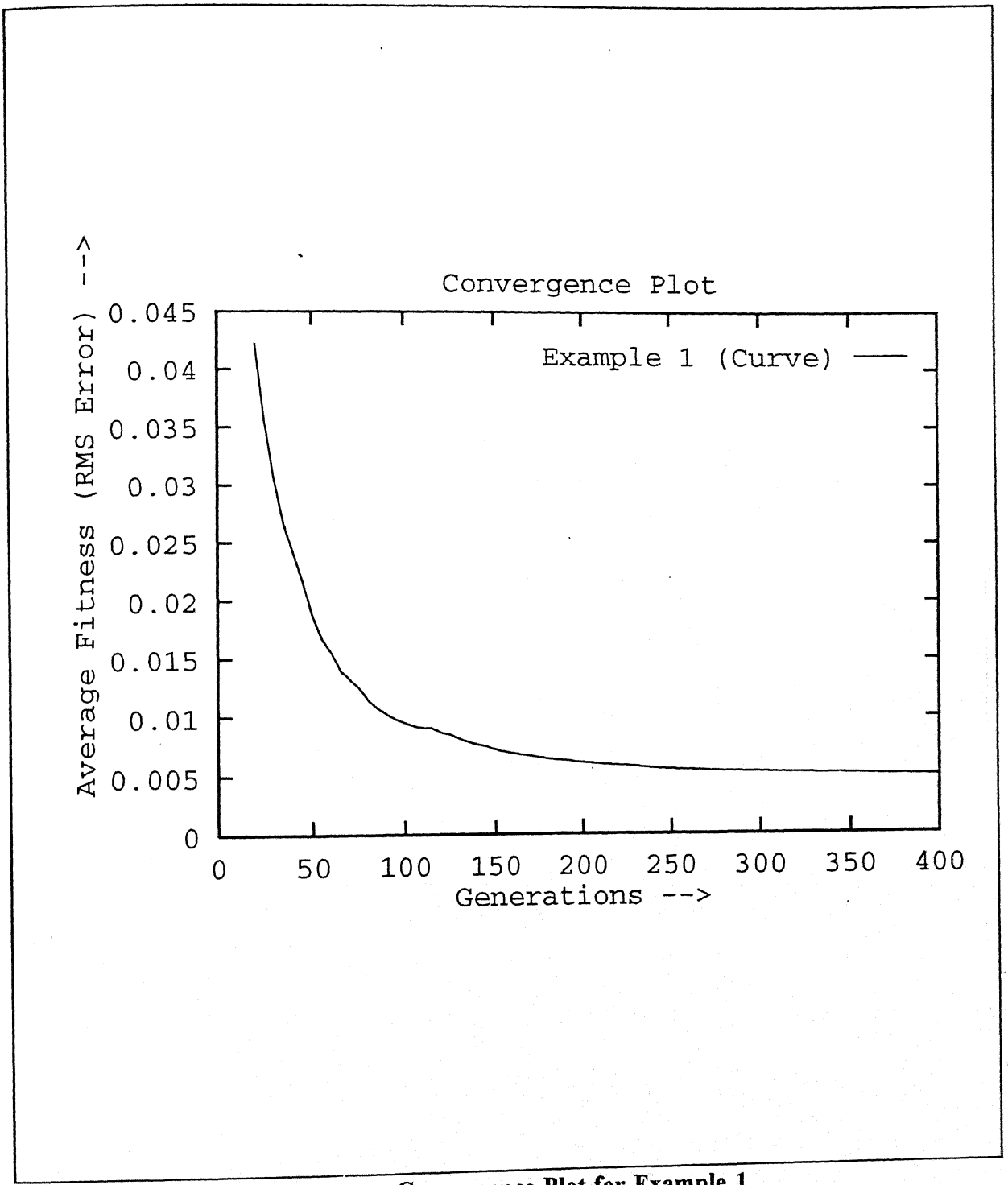
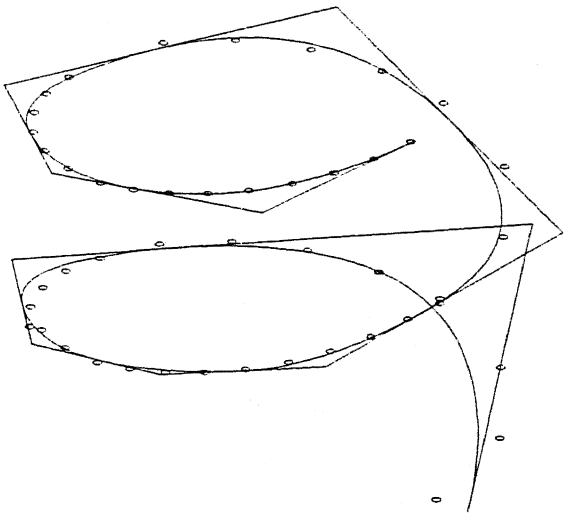
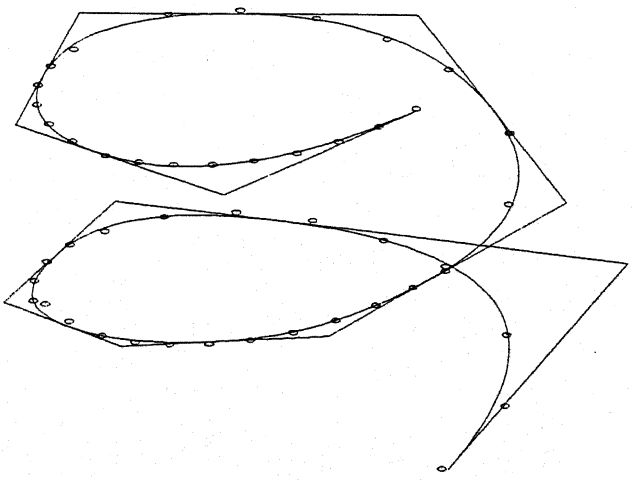


Figure 3.2 Convergence Plot for Example 1



a. Before Optimization



b. After Optimization

Figure 3.3 B-Spline Curve Fit for Example 2

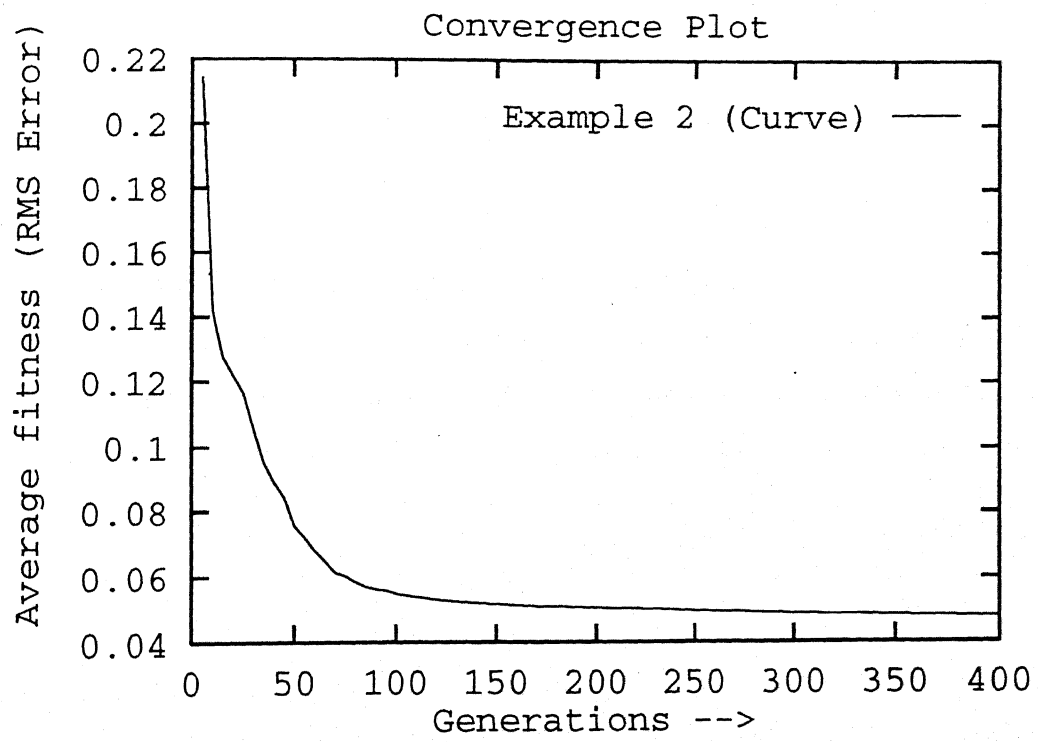


Figure 3.4 Convergence Plot for Example 2

Chapter 4

GA Based Method for Surface Fitting

This chapter deals with a GA based method for B-spline surface fitting. The objective is to fit a B-spline surface through the given data points such that the fitting error is minimum. The order of the surface in u and v direction and the number of control points are to be specified by the user. To ascertain the performance of the method the points from a mathematically known surface are taken as input.

The initialization scheme used for u parameter in curve fitting (Section 3.1, Chapter 3) has been extended to both parametric directions u and v. The approach for surface fitting differs from curve fitting in the way the knot vectors are generated. Basically two knot vectors, one each in u and v parametric direction are needed. Section 4.3 discusses the details of the knot selection procedure.

4.1 Parameterization Model

Given an ordered set of grid points, P_i , the parameter values u_i and v_i can be found by any of the parameterization method, namely uniform, chord length or the centripetal. These methods are discussed in Section 2.2 of Chapter 2.

A generalization of the above three schemes is proposed by Lee[12] as,

$$u_{ij} = \frac{\sum_{l=1}^{i-1} |P_{l+1,j} - P_{l,j}|^e}{\sum_{l=1}^{s-1} |P_{l+1,j} - P_{l,j}|^e} \qquad (4.1)$$

$$2 \le i \le s \; ; \; 1 \le j \le r$$

$$v_{ij} = \frac{\sum_{l=1}^{j-1} |P_{i,l+1} - P_{i,l}|^e}{\sum_{l=1}^{r-1} |P_{i,l+1} - P_{i,l}|^e} \qquad (4.2)$$

$$1 \le i \le s \; ; \; 2 \le j \le r$$

where the data points constitute the grid of *rxs* points. At e = *0, .5, 1* the expression reduces to uniform, centripetal, and chord length parameterizations respectively. The population initialization scheme has been based on the above model. If *popsize* is the number of populations and *rxs* points are specified on the surface , then a generic population composed of u_{lij} and v_{lij} is given by

for (l=0 to popsize)

{

e= random number between 0 and 1

find out u_{lij} and v_{lij} using Equation 4.1 and 4.2 respectively.

}

This initialization ensures that the populations of u_{ij} and v_{ij} will be set of monotonically increasing

real numbers with additional constraints

$0 <= u_{lij} < X_{nu+ku}$ and $0 <= v_{lij} < Y_{nv+kv}$

X_{nu+ku} and Y_{nv+kv} represent maximum value of knot vector in u and v directions respectively.

4.2 Objective Function Formulation

Let P_i be the measured data points, supplied by the user and P_i^c be the points on the interpolated B-spline surface evaluated at the points corresponding to the parameter u_i and v_i.

Then ,

$$[P^c] = [C][[C]^T[C]]^{-1}[C]^T[P] \tag{4.3}$$

Error in x, y, z coordinates can be stated as

$$\begin{aligned} \delta x &= \left|P_x^c - P_x\right| \\ \delta y &= \left|P_y^c - P_y\right| \\ \delta z &= \left|P_z^c - P_z\right| \end{aligned} \tag{4.4}$$

Euclidean error expression at a particular point will be

$$e_i = \sqrt{\delta x^2 + \delta y^2 + \delta z^2} \tag{4.5}$$

Expressions for total error and RMS error will respectively be

$$e_{total} = \sum_{i=1}^{r*s} e_i \tag{4.6}$$

$$e_{rms} = \sqrt{\sum_{i=1}^{r*s} \{\frac{e_i}{r*s}^2\}} \tag{4.7}$$

Optimization problem is formulated as " minimize RMS error where the design variables are parameter values u_i and v_i and the variables are bounded between 0 and the maximum value of the knot vector X_{nu+ku} and Y_{nv+kv} respectively".

4.3 Knot Vector Selection

In the case of surface fitting, two knot vectors for u and v parametric direction have to be generated. Similar to the case of curve fitting here also average knot scheme (Appendix B) is used to generated the knot vector. Unlike the case of curve fitting here exist a set of u and v values corresponding to each characteristic curve in u and v direction. For eg., If a grid of $6x5$ points is given, there will be six characteristic u curves and five characteristic v curves.The knot selection is made based on the fitness ranking of the population. Recalling that the initial population was created using $e \in [0,1]$, the initial population is evaluated corresponding to $e=0$, $.25$, $.5$, 1. For each of these e values, parameter values for all characteristic u curves are sorted in an ascending array. Likewise, parameter values for all characteristic v curves are also sorted in ascending array. The average knot scheme is now applied to these two sorted arrays. The population having the best fitness among these four, is used to generate the knot vectors.

4.4 Implementation Details

The algorithm of the method can be summarized as follows

Initialize population using the exponential initialization scheme;

Evaluate the fitness corresponding to e=0,.25,.5,1 to decide upon the population, to generate knot vector;

Sort the parameter values of all u characteristic curves in ascending order;

Sort the parameter values of all v characteristic curves in ascending order;

Find the knot vector using average knot vector scheme;

(while termination criteria are unsatisfied){

Reproduction ;

Crossover (using SBX strategy);

Mutation;

Evaluation;

}

4.5 Computational Aspects :

Software has been developed based on above formulation . First fit solution can be obtained by B-spline surface fitting technique (Section 2.1.2 (b)). Subsequent optimization is carried out using real coded GA with SBX crossover (Appendix C). The termination criteria is the reduction in error value or the maximum number of generations. The inversion of the blending function matrix is carried out using Cholesky decomposition method. The graphical output is created in the form of script files.

4.6 Examples and Discussions

Two examples are provided for the case of B-spline surface fitting. The first example is for singly curved surface and the second example is for doubly curved case. The input data points have been taken from the surfaces having known mathematical form. The surfaces are fitted using the user supplied information regarding the number of defining polygon vertices, the order of the surface in both parametric directions, and the information needed for the optimization run i.e., population size, the maximum number of generations , probability of crossover, probability of mutation and the distribution index. The uniform crossover on all the design variables is implemented.

Example 1: Singly Curved Surface

In this example the input points have been taken from a surface which is having curvature only in one parametric direction. The number of design variables in this case are equal to the number of u as well as the v parameter values i.e., 98 in this case. The input data and the results of the optimization run are shown in Table 4.1 . The first fit result of the surface deviate considerably from the input data points. The final fit obtained after the optimization run conform to the data points to a reasonable level of accuracy. The Figure 4.1 and 4.2 show the fitting results. Figure 4.3 shows the convergence plot . At 200^{th} generation the error becomes almost constant.

Input		Output	
Number of Measured Points	7x7 =49	Initial Error	Final Error
Number of Polygon Points	5x5 =25	0.109	0.064
Order in u Direction	3		
Order in v Direction	3		
Population Size	50		
Number of Generations	200		
Probability of Crossover	1		
Probability of Mutation	0.01		
Distribution Index for SBX	2		

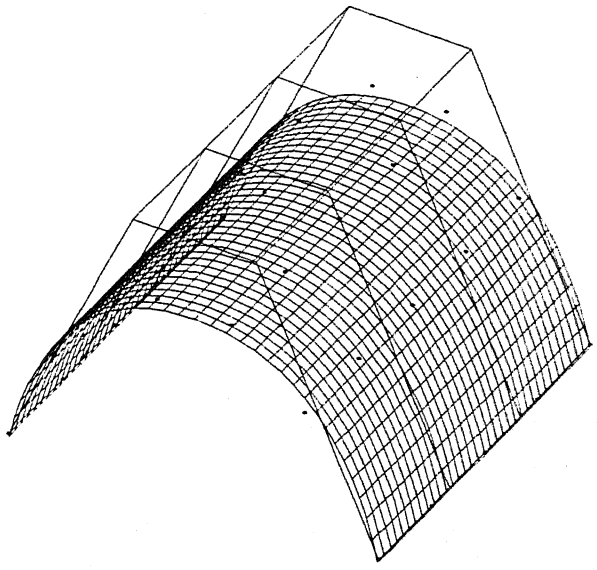
Table 4.1 Numerical Data for Example 1

Example 2: Doubly curved surface

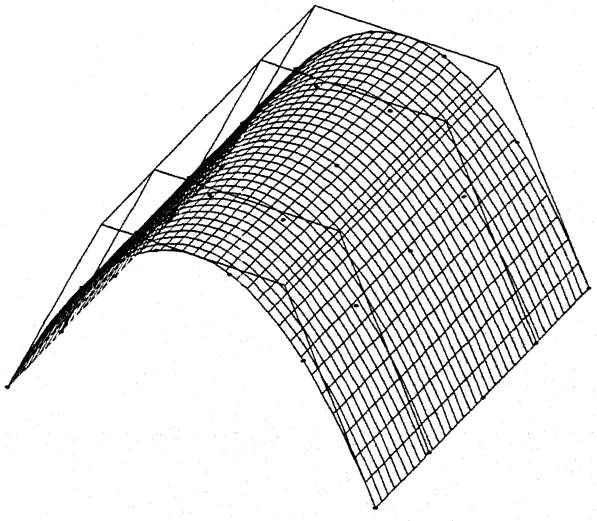
Input data for Example 2 has been taken from a doubly curved surface. The number of measured points are 100. Hence the design variables become 200. Figure 4.4 shows the first fit results and the final fit result. Table 4.2 contains the input and output data and Figure 4.5 shows the convergence plot.The error becomes almost constant after 600^{th} generation.

Input		Output	
Number of Measured Points	10x10 =100	Initial Error	Final Error
Number of Polygon Points	5x5 =25	0.4948	0.0267
Order in u Direction	3		
Order in v Direction	3		
Population Size	60		
Number of Generations	800		
Probability of Crossover	1		
Probability of Mutation	0.01		
Distribution Index for SBX	2		

Table 4.2 Numerical Data for Example 2



a. Before Optimization



b. After Optimization

Figure 4.1 Fitting Results for Example 1

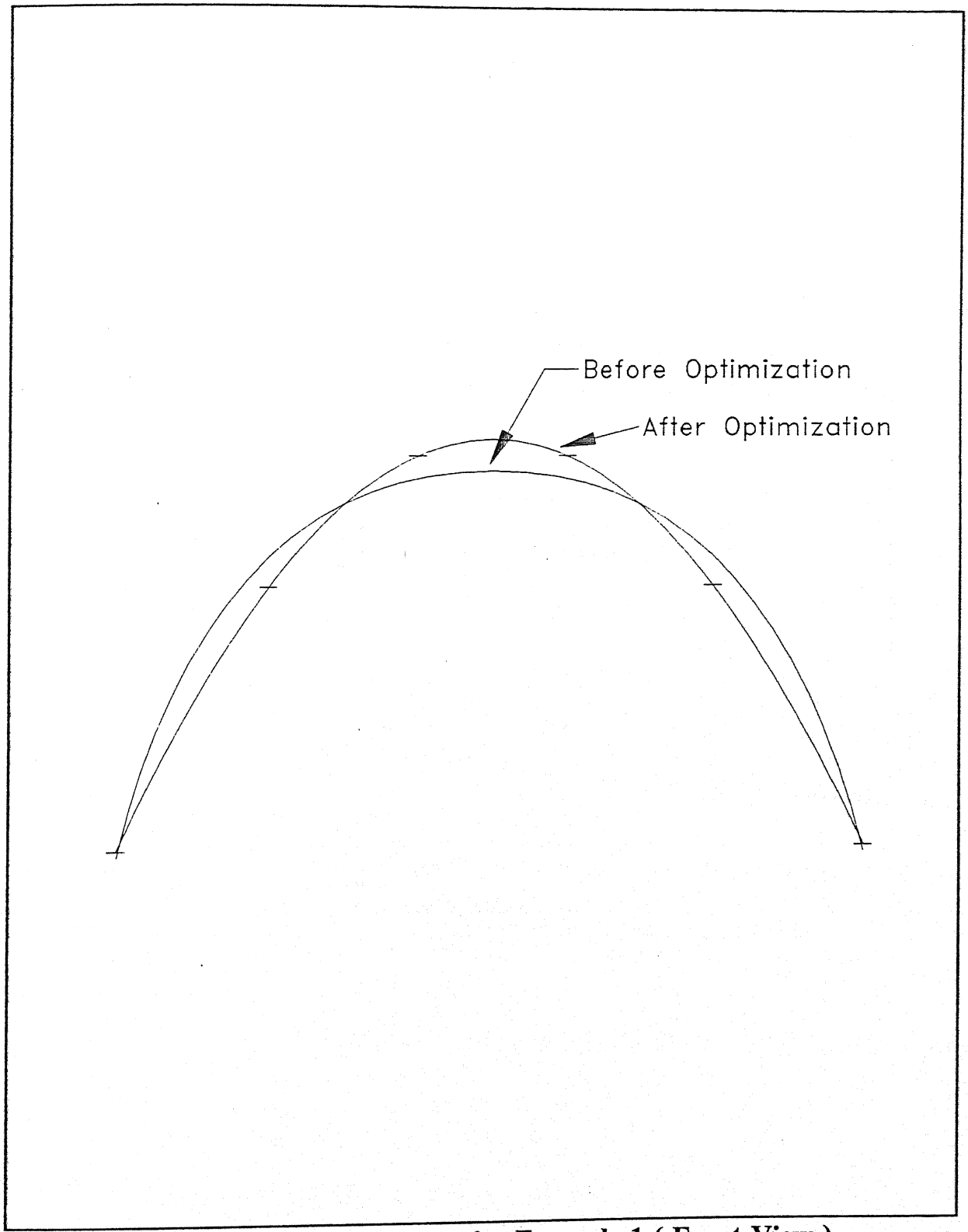


Figure 4.2 Fitting Results for Example 1 (Front View)

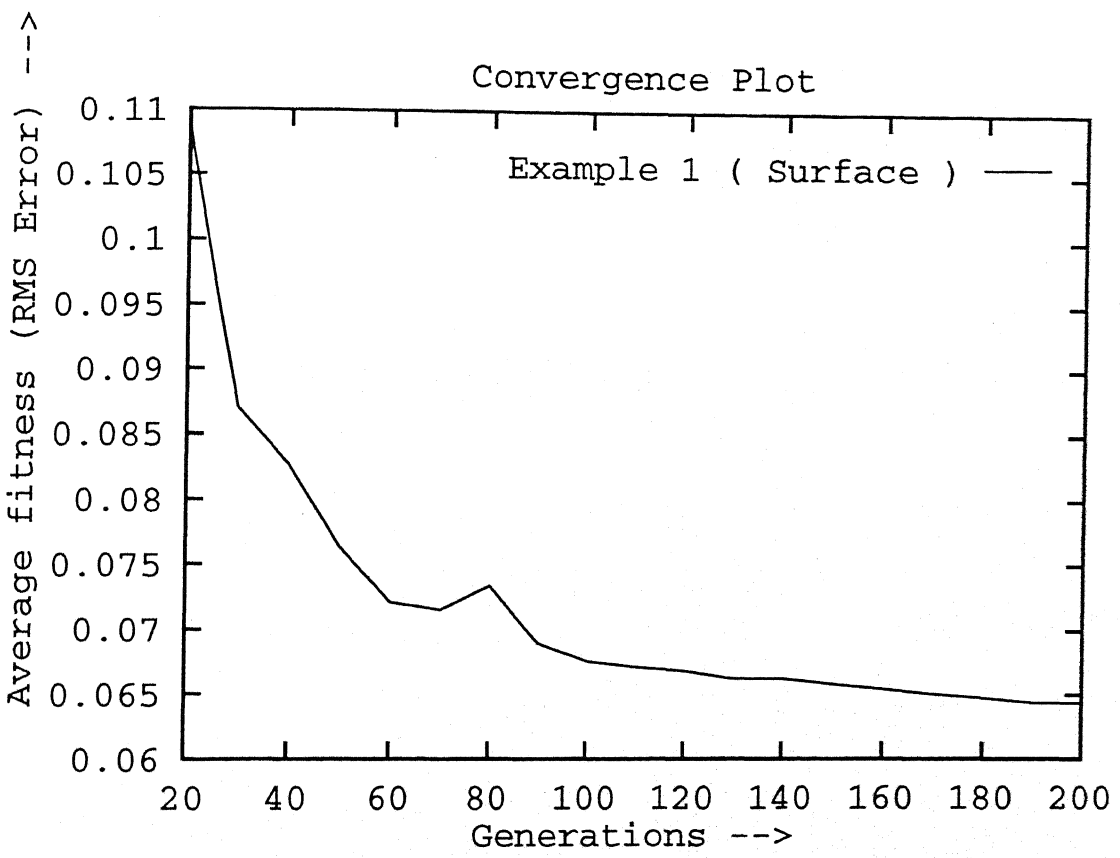


Figure 4.3 Convergence Plot for Example 1

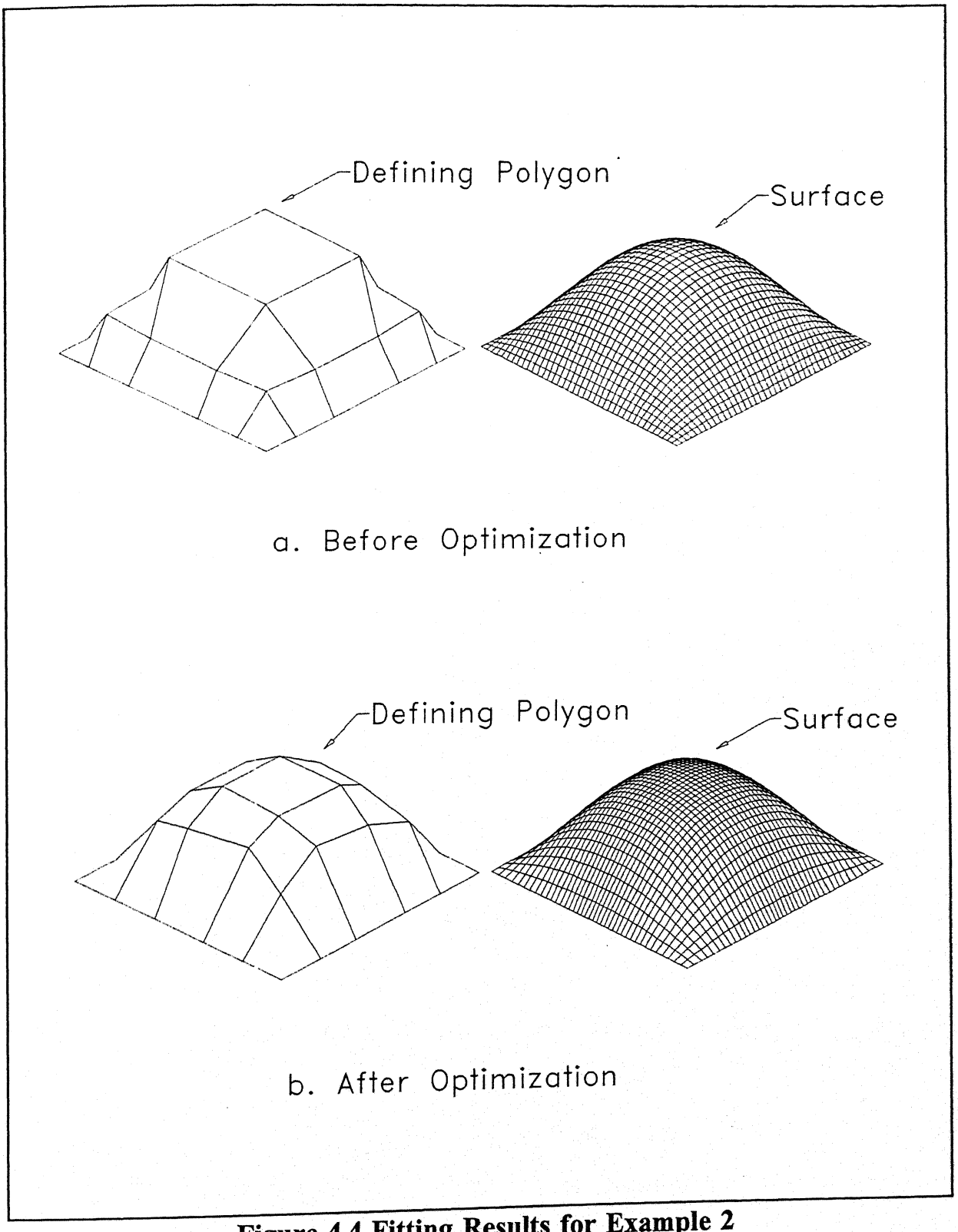


Figure 4.4 Fitting Results for Example 2

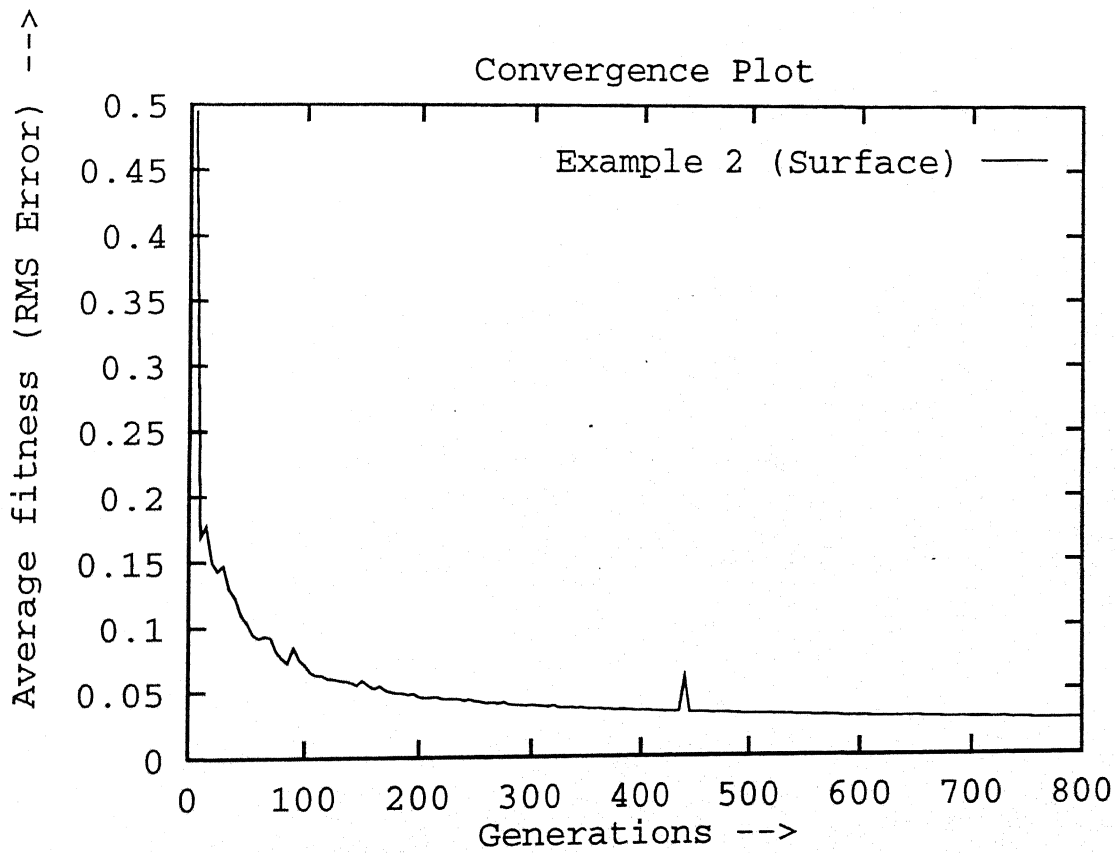


Figure 4.5 Convergence Plot for Example 2

Chapter 5

Surface Localization and Error Evaluation

Part localization refers to the problem of determining the position and the orientation of an object with respect to its CAD database [6]. Associated with the CAD database description is a coordinate frame referred to as CAD or the part frame; This is the frame in which the part is designed and viewed on the screen of the CAD station. Generalizing the above definition for surfaces, the surface localization problem can be stated as matching of the functional surface with its surface model.

In Section 1.3 of Chapter 1, various activities of a reverse engineering system have been discussed. It is noteworthy that the third activity in the above framework, namely postprocessing, invariably introduces surface form errors. This is especially true when the prototyping process is rapid prototyping. The errors may arise because of the mechanism of the system optics, residual stresses induced by initial curing, post curing or in the finishing operations [9]. Surface localization can be used to evaluate the deviations of the 'prototype surface' with the 'design surface' which is obtained in the fitting stage.

5.1 Error Evaluation

The error evaluation is based on the comparison of the measured data from the prototype surface with the design surface. Though the design surface is having parametric representation, the comparison is essentially at the point level. There are two types of errors affecting the dimensional accuracy of the prototypes.

1. Form error, which is introduced in the prototyping process.
2. The offset error involved in tilting and translation of the prototype surface with respect to coordinate reference frame of the design surface (Figure 5.2).

Generally, the dimensional errors of the sculptured surface are the sum of the form error and the

offset error (Figure 5.1) [8].

An optimal surface localization scheme has been implemented in the present work to evaluate the form error and the offset.

5.2 Mathematical Formulation

An object possessing three orthogonal datum faces can be positioned unambiguously in space by traditional 3-2-1 positioning technique [6]. Here, a total of six measurements are taken in the sequence of 3, 2 and 1 measurements on three orthogonal faces (Figure 5.2). In the case of free-form prototype surface, which in general will be having form error as well as the offset error six measurement points can not serve the purpose. More over to evolve some measure of form error the measurement points should be sufficient in number. This number may vary depending on the surface form. In the present application, the same number of measured points are used which were used previously for the surface fitting. So, the problem of matching the two surfaces is generally overspecified [7].

Mathematically,

$$Total\ Error = [P] - [Q^P][T] \tag{5.1}$$

where $[Q^P]$ is the set of measured points on prototype surface and [P] are the corresponding points on the design surface. [T] is the general transformation matrix and is a function of l, m, n, α, β, Υ which are the parameters of rigid body motion , namely three translation and three rotation parameters about x,y,z axes respectively.

Here the point to be noted is correspondence of a corner of the prototype surface to that of design surface is assumed known. This assumption is justified in most of the manufacturing situations.

$$[T] = [T_{Rx}][T_{Ry}][T_{Rz}][T_{Tr}] \tag{5.2}$$

T_{Rx}, T_{Ry}, T_{Rz} are the three rotation matrices about x, y, and z axes respectively. T_{Tr} is the translation matrix. The $[Q^P][T]$ represents a generic instance of the prototype surface in space.

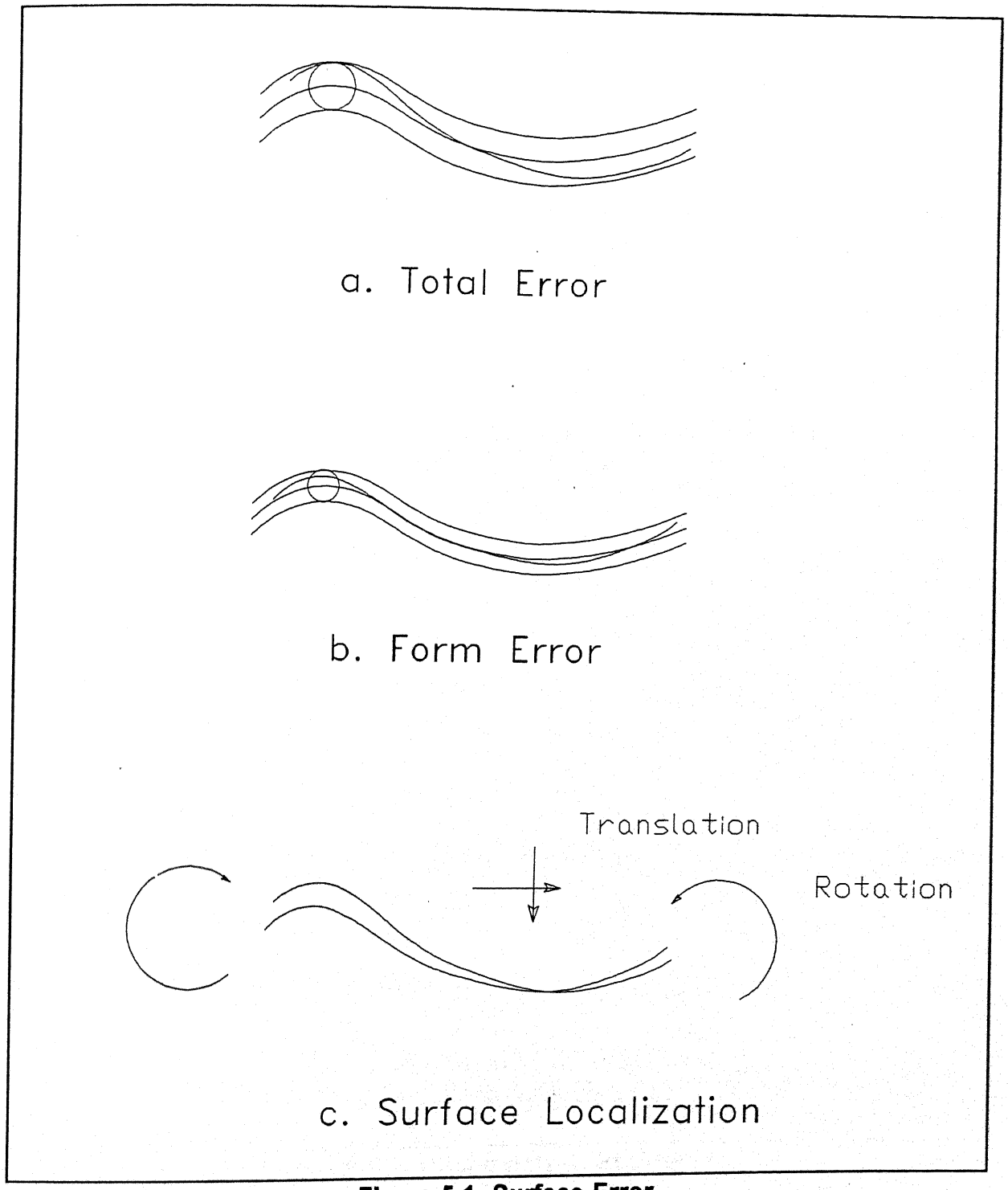
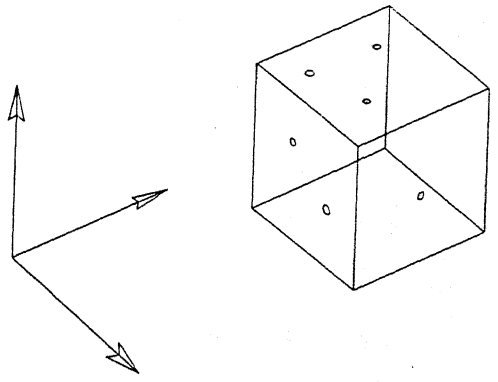
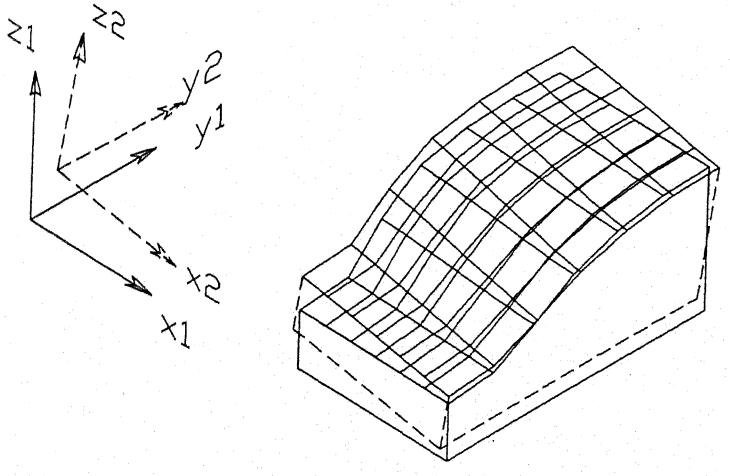


Figure 5.1 Surface Error



a. Traditional 3–2–1 Positioning



b. Uncertainity in Positioning
of the Prototype Surface

Figure 5.2 Uncertainty in Positioning

$$[T_{Rx}] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha & 0 \\ 0 & -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.3}$$

$$[T_{Ry}] = \begin{bmatrix} \cos\beta & 0 & -\sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.4}$$

$$[T_{Rz}] = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 & 0 \\ -\sin\gamma & \cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.5}$$

$$[T_{Tr}] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ l & m & n & 1 \end{bmatrix} \tag{5.6}$$

The problem has been formulated as

$$e_{rms} = MIN \left\{ \sum_{i=1}^{N} \left\{ \frac{\{P_i - Q^p{}_i[T]\}^2}{N} \right\} \right\}^{\frac{1}{2}} \tag{5.7}$$

where N refers to the total number of measured points. The objective is to minimize the error expression. The design vector is given by {l, m, n, α, β, Υ}. Here l, m, n represent translations along x,y and z directions and α, β, Υ represent the rotations about x, y and z axes respectively.

5.3 Implementation Details

The problem posed in the preceding section involves trigonometric functions. The Nelder and Mead's simplex search method is used as the search technique. It is a population based direct search technique requiring only the information about the function value. The technique uses simple operators like reflection, expansion, contraction operators (Appendix D) to determine the transition rule.

The simulated experiments are carried out to ascertain the effectiveness of the approach. Two types of perturbation are applied to simulate the effect of the offset error and the form error respectively.

1. Arbitrary transformation is applied to design surface to evolve the simulated prototype surface.
2. Each point of the surface so obtained is perturbed with some probability to simulate the effect of the form error.

The optimization routine is started with design surface and the prototype surface. As the optimization proceeds the prototype surface is oriented optimally with respect to the design surface. At the optima, the value of rms error represents form error and the transformation matrix evolved indicates the offset of the prototype surface with design surface.

5.4 Examples and Discussions

In the examples given below trial vector represents the initial guess of the design vector needed to start the simplex; apart from rms error local maximum error between the two surfaces is also evaluated. The offset vector represents the amount of offset present in the prototype surface with respect to design surface.

Example 1: Two set of numerical experiments are carried out as explained below

Case 1.) Only offset perturbation is applied to design surface to generate fictitious prototype surface. As no form error is present in the surface the error expression should reduce to zero. Table 5.1 and Figure 5.3 show the results of the program run. The results justify the above logic. The Figure 5.4 shows the convergence plot for the above case.

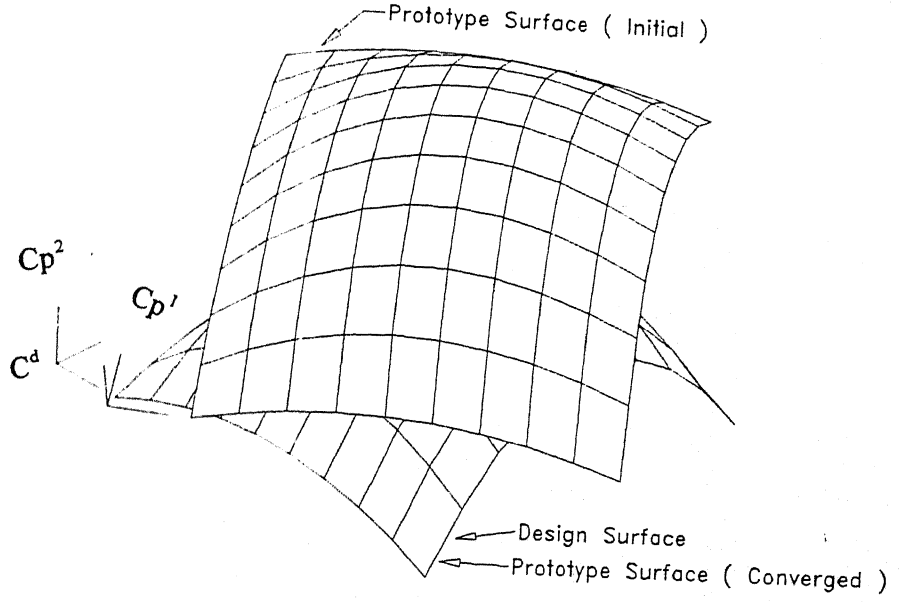
Input		Output	
Measured points	100	RMS error	6×10^{-6}
Offset perturbation	[6, 4, 17, 23, 10, 56]	Local maximum error	$1x\ 10^{-6}$
Trial design vector	[12,3, -4, -23, 12, 10]	No. of function evaluations	1972
		offset vector	[-7.43, -5.72, -15.9, -5.408, -24.415, -52.788]

Table 5.1 Numerical Data (Example 1, Case 1.)

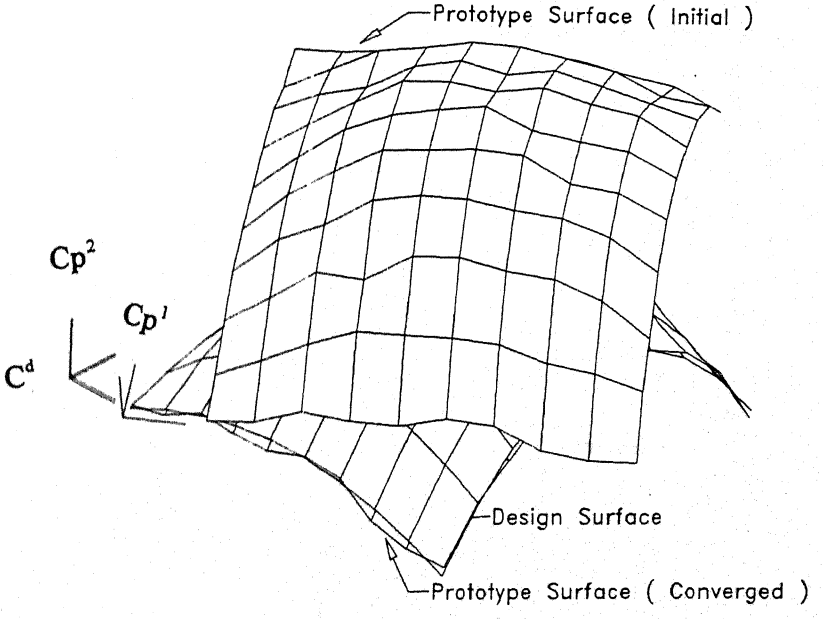
Case 2.) Both offset as well as the form perturbations are applied. At optima, the RMS error represents the form error and the offset vector charecterizes the offset of the prototype surface with the design surface. Table 5.2 , Figure 5.3 and 5.4 show the results.

Input		Output	
Measured points	100	RMS error	0.28126
Offset perturbation	[6, 4, 17, 23, 10, 56]	Local maximum error	0.654667
Trial design vector	[12, 3, -4, -23, 12, 10]	No. of function evaluations	1600
		offset vector	[-7.4015, -5.713, -16.0328, -5.36, -24.569, -52.768]

Table 5.2 Numerical Data (Example 1, Case 2.)

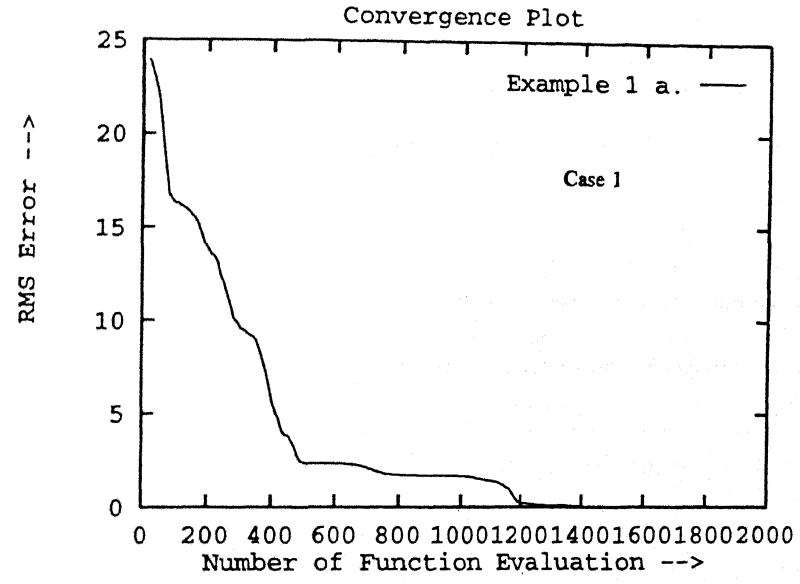


Case 1.



Case 2.

Figure 5.3 Surface Localization Results (Example 1)



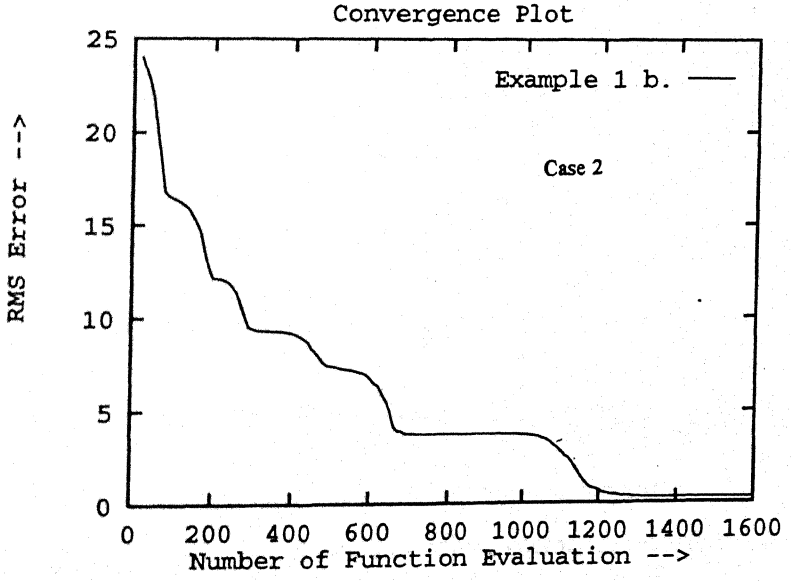


Figure 5.4 Convergence Plots (Example 1)

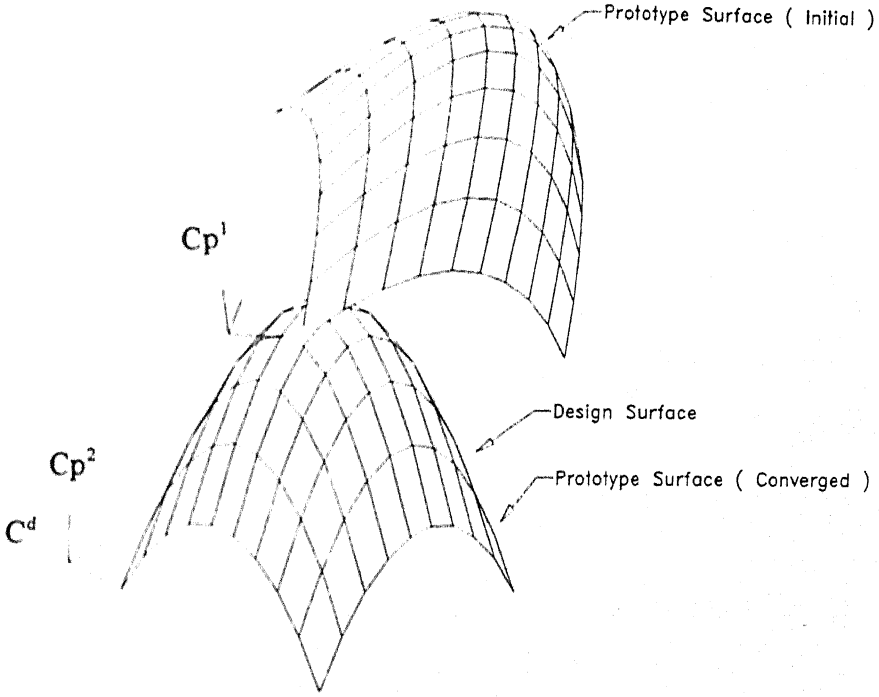
Example 2 :

Case 1.) Only offset perturbation is applied. The error expression almost converges to zero. The results are shown in Table 5.3 , Figure 5.5 and Figure 5.6 .

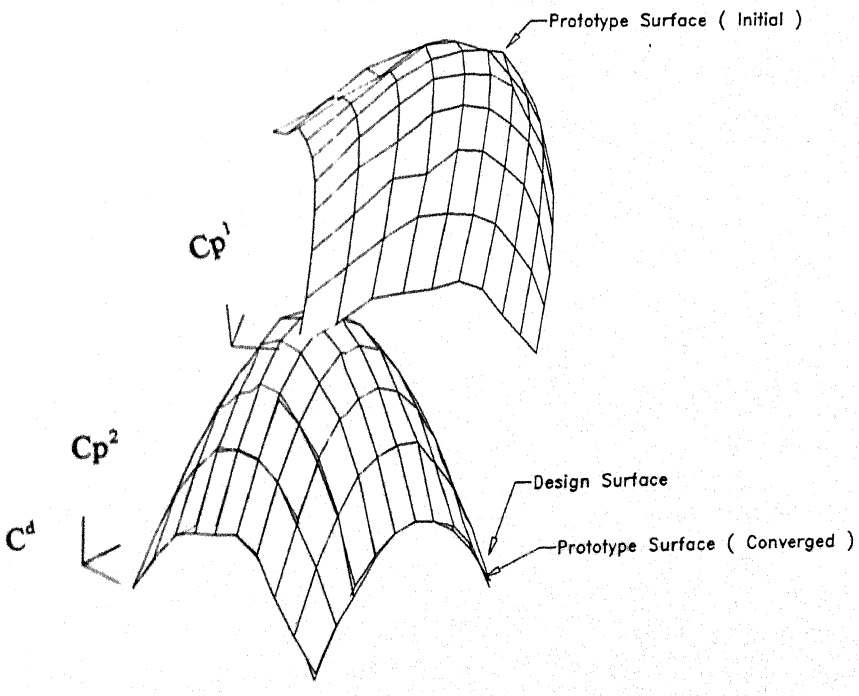
Input		Output	
Measured points	100	RMS error	$2x10^{-6}$
Offset perturbation	[5, -5, 6, 23,0, 34]	Local maximum error	$6x10^{-6}$
Trial design vector	[2, 2, 2, 2, 2, 2]	No. of function evaluations	2188
		offset vector	[-8.03, 2.96, -3.568, -19.384, -12.624, -31.834]

Table 5.3 Numerical Data (Example 2, Case 1.)

Case 2.) Both offset as well as the form perturbation are applied. At the optima RMS error expression represents the form error and the offset vector represents the offset of the prototype surface with the design surface. The results are shown in Table 5.4 , Figure 5.5 and 5.6 .

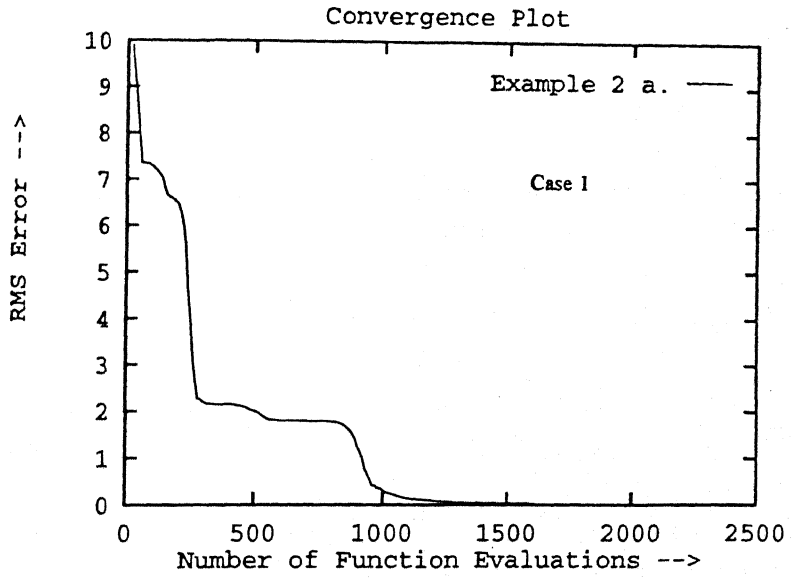


Case 1.



Case 2.

Figure 5.5 Surface Localization Results (Example 2)



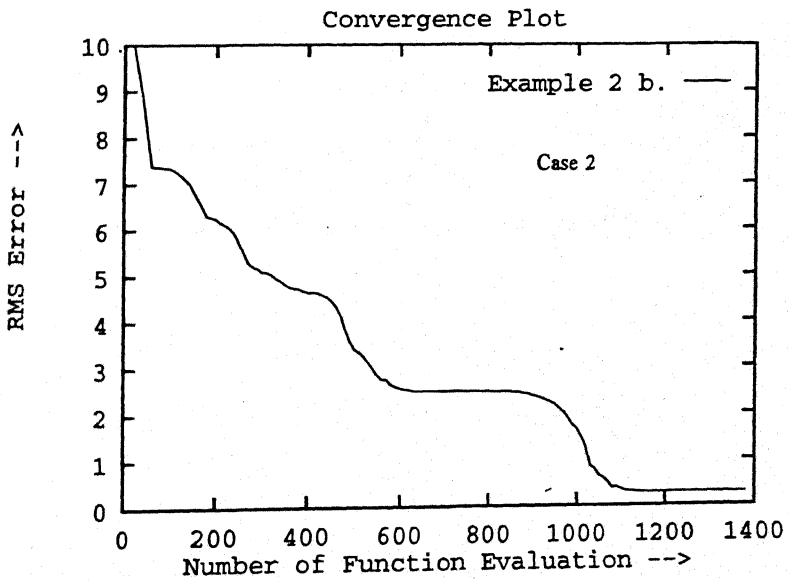


Figure 5.6 Convergence Plots (Example 2)

Input		Output	
Measured points	100	RMS error	0.2802
Offset perturbation	[5, -5, 6, 23,0, 34]	Local maximum error	0.6739
Trial design vector	[2, 2, 2, 2, 2, 2]	No. of function evaluations	1387
		offset vector	[-8.003, 2.9606, -3.69, -19.418, -12.854, -31.7909

Table 5.4 Numerical Data (Example 2, Case 2.)

Chapter 6

Conclusions

6.1 Technical Summary

GA based B_Spline curve and surface fitting, and optimal surface localization schemes have been developed and implemented in the present work. Both of these methods find their use in ' Reverse Engineering '. Suitable examples have been provided to illustrate the effectiveness of the approach. Chapter 1, outlines the framework of a reverse engineering system and the various activities involved. Among these activities ' Fitting ' and ' Surface localization' are the scope of the present work. Chapter 2, covers the mathematical formulation of the B-Spline curve/surface fitting and discusses ' Real Coded GAs ' in the context of its applications. Implementation details of the GA based B-Spline curve fitting have been described in Chapter 3. Two illustrative examples are provided; first one is meant for the fitting of a planar curve and the second one deals with the space curve.Chapter 4, covers the implementation details of the GA based method for B-Spline surface fitting. Two examples for the case of singly curved and the doubly curved surface fitting are provided. Optimal surface localization scheme and its use in error evaluation of the prototype surface are the subject matter of the Chapter 5. Two examples illustrate the approach. Software has been developed for the above mentioned methods in C programming language under UNIX environment. The graphical interface has been provided by using the ' Script File ' concept of AutoCAD.

GA based method for curve and surface fitting, proposed in this work has got distinct advantages in terms of search power over the iterative fitting schemes of Rogers & Fog [2] and Hoschek [5]. The results obtained are encouraging and they indicate a stable convergence pattern for the fitting problem. The difficulties associated with the invertibility of blending function

matrix, have been overcome by implementing average knot scheme, suggested by Ma & Kruth [3].

Typically, the goal of an optimization process is not only to achieve an optimal solution but also to find an optimal process to achieve the goal. To enhance the effectiveness of the search, it is always desirable if we can supply some useful information regarding the problem. Population initialization scheme presented in this work, generates this additional information. Overall reduction in search time is realized by the above said implementation.

Second problem, namely surface localization will be useful in accurate measurement of the form error in the case of free-form surfaces.

Lastly, the two problems studied in the above work will be very useful in the development of an integrated reverse engineering system. Typically, the objective of an integrated reverse engineering system is to obtain a prototype starting from a physical object, within a short time span with minimal of human interference .

6.2 Suggestions for the Future Work

- The method developed for B-Spline curve and surface fitting can be extended to cover closed curves and surfaces as well by making use of periodic basis functions.
- At present the surface fitting method is meant for single surface patch. Algorithms can be developed using which several such patches can be combined or stitched with some continuity requirement imposed on them.
- The method has been evaluated for simulated set of data points. Physical experiments can be carried out using CMM and the rapid prototyping machine to judge its performance.
- The surface localization scheme presented in this work uses simplex search method. For surfaces having lot of waviness some global optimization technique like GAs or simulated annealing should be implemented.
- Lastly, the present work addresses only two activities of reverse engineering system, namely ' fitting' and the ' surface localization ' . Methodologies for the other two activities, scanning and the postprocessing (NC code generation or .STL file creation) can be developed in order to

realize an integrated reverse engineering system. Scanning procedure can be automated by programming the coordinate measuring machine (CMM). When the object to be scanned is sculptured surface, the collision free path generation is an important area of research. The surface obtained after fitting can be used for NC code generation if the prototyping process is CNC machining. Alternatively, some slicing software can be developed , which will convert the surface definition to .STL files, which in turn will serve as the input to a rapid prototyping machine.

References

[1] Green, P.A., and Philpott, M.L.,(1994), " Error Modeling of Reverse Engineered Free Form Surfaces ",*Transactions of NAMRI/SME*,vol.22,1994,pp.259-266.

[2] Rogers, D.F., and Fog, N.G.,(1989), " Constrained B-spline Curve and Surface Fitting",*Computer Aided Design*,vol. 21,No. 10,pp. 641-648.

[3] Ma, W., and Kruth, J.P.,(1995), " Parameterization of Randomly Measured Points for Least Square Fitting of B-Spline Curves and Surfaces",*Computer Aided Design* ,vol. 27,No. 9,pp. 663-675.

[4] Sarkar, B., and Menq, C.H.,(1991),"Smooth Surface Approximation and Reverse Engineering",*Computer Aided Design*,vol. 23,No. 9,pp. 623-628.

[5] Hoschek, J.,(1987),"Intrinsic Parameterization for Approximation",*Computer Aided Geometric Design*, Vol. 5,pp. 27-31.

[6] Gunnarson, K.T., and Prinz, F.B.,(1987)," CAD Model Based Localization of Parts in Manufacturing "*Computer*, pp. 66-73.

[7] Pahk, H.J., Kim, Y.H., Hong, Y.S., Kim, S. G.,(1993),"Development of Computer-Aided Inspection System with CMM for Integrated Mold Manufacturing", *Annals of the CIRP*,Vol. 42/1/1993,pp. 557-560.

[8] Shunmugam, M.S.,(1991),"Criteria for Computer Aided Form Evaluation",*Journal of Engineering for Industry*, Vol. 113,pp. 233-238.

[9] Philpott, M.L., and Green, P.A.,(1995), "An Error Compensation Strategy for Replication by

Rapid prototyping",*Journal of Engineering for Industry*, Vol. 117,pp. 423-429.

[10] Reckoff, M.G.,(1985), "On Reverse Engineering",*IEEE Transaction Systems, Man and Cybernetics*, pp. 244-252.

[11] Feng, S.C., and Duffie, N.A.,(1985), "Modification of Bicubic Surface Patches Using Least Square Fitting Techniques",*Computers in Mechanical Engineering*, Sept. ,pp. 57-65.

[12] Lee, E.T.,(1989), " Choosing Nodes in Parameteric Curve Interpolation", *Computer Aided Design*, vol. 21,pp 363-370.

[13] Faux ,I.D., and Pratt, M.J.,(1983),*Computational Geometry for Design and Manufacture*, Ellis Horwood, Chicester.

[14] Rogers, D.F., and Adams, J.A.,(1990), *Mathematical Elements for Computer Graphics*, McGraw-Hill, New York.

[15] Zeid, I.,(1991), *CAD/CAM Theory and Practice*, McGraw-Hill,New York.

[16] Farin, G.,(1990), *Curves and Surfaces for Computer Aided Geometric Design*, Academic Press, Inc.

[17] Deb, K.,(1995) *Optimization for Engineering Design: Algorithms and Examples*, Prentice-Hall ,India.

[18] Goldberg,D.E.,(1989) Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley.

[19] Agrawal, R. B.,(1995), " *Simulated Binary Crossover for Real - Coded Genetic Algorithms: Development and Application in Ambiguous Shape Modelling* ", Master's Thesis, Department of Mechanical Engineering, Indian Institute of Technology, Kanpur.

Appendix A

Invertibility Issue of Matrix $[C]^T[C]$

A.1 Schoenberg-Whitney Conditions for Curve Fitting

Let $\{N_i\}^n$ be the n normalized B-Splines defined over knots $X=\{X_i\}^{n+k}$ and $u=\{u_i\}^m$ be the location parameters. The matrix C defined by equation 2.12 for curve fitting is of full column rank if and only if there exists $u' = \{u_i\}^n \subset u$ such that

$$N_i(u_i') \neq 0 \qquad i = 1,2,3,...,n \tag{A.1}$$

or

$$X_i < u_i' < X_{i+k} \qquad i=1,2,3,...,n \tag{A.2}$$

In the case of presence of k-multiple knots over the definition domain of N_i ,the i^{th} condition for u_i' of equation A.2 can be relaxed to

$$X_i \leq u_i' < X_{i+k} \qquad X_i = ... = X_{i+k-1} < X_{i+k} \tag{A.3}$$

$$X_i < u_i' \leq X_{i+k} \qquad X_i < X_{i+1} = ... = X_{i+k}$$

A.2 Schoenberg-Whitney Conditions for Surface Fitting

Let $\{N\}^{nu}$ *and* $\{M_j\}^{nv}$ be the normalized B-Splines defined over the knots $X=\{X_i\}^{nu+ku}$ and $Y=\{Y_j\}^{nu+kv}$, and $u=\{u_i\}^m$ and $v=\{v\}^m$ be the location parameters.The matrix C defined by equation 2.18 for surface fitting is of full column rank if and only if there exists $u'=\{\{u_{ij}'\}^{nu}\}^{nv} \subset u$ and there exists $v'=\{\{v_{ij}'\}^{nu}\}^{nv} \subset v$ such that

$$N_i(u'_{ij}).M_j(v'_{ij}) \neq 0 \quad i = 1,2,...,nu \;\; ; \;\; j = 1,2,...,nv \tag{A.4}$$

or

$$\begin{aligned} X_i < u'_{ij} < X_{i+ku} \\ Y_i < v'_{ij} < Y_{i+kv} \end{aligned} \quad i = 1,2,...,nu \quad j = 1,2,...,nv \tag{A.5}$$

In the case of presence of ku -multiple knots over the definition domain of N_i, the nv conditions for u_{ij}' of expression A.5 can be relaxed to

$$\begin{aligned} X_i \leq u'_{ij} < X_{i+ku} \qquad & X_i = ... = X_{i+ku-1} < X_{i+ku} \\ X_i < u'_{ij} \leq X_{i+ku} \qquad & X_i < X_{i+1} ... = X_{i+ku} \end{aligned} \tag{A.6}$$

$$j=1,2,...,nv$$

where $j = 1,2,...,nv$. In the case of presence of kv -multiple knots over the definition domain of M_i, the nu conditions for v_{ij}' of expression A.5 can be relaxed to

$$\begin{aligned} Y_j \leq v'_{ij} < Y_{j+kv} \qquad & Y_j = ... = Y_{j+kv-1} < Y_{j+kv} \\ Y_j < v'_{ij} \leq Y_{j+kv} \qquad & Y_j < Y_{j+1} = ... \; Y_{j+kv} \end{aligned} \tag{A.7}$$

$$i=1,2,...,nu$$

Appendix B

Average Knots

Let u=$\{u\}^m$ be a set of nondecreasing location parameters allocated ,

$$u_i \le u_{i+1} \qquad 1 \le i \le m-1 \tag{B.1}$$

The average knots are defined as

$$X_i = \begin{cases} u_1 & 1 \le i \le k \\ \mu_i + \sum_{j=j_1+1}^{j_2} u_j & k+1 \le i \le n \\ u_m & n+1 \le i \le n+k \end{cases} \tag{B.2}$$

where

$$\mu_i = u_{j_c} + r \cdot (u_{j_c+1} - u_{j_c}) \tag{B.3}$$

$$j_1 = j_c - h - (1 - \delta) \cdot int\left(\frac{3(1-r)}{2}\right) \tag{B.4}$$

$$j_2 = j_c + h - (1 - \delta) \cdot int\left(\frac{3(1-r)}{2}\right) \tag{B.5}$$

with

$$h = int\left(\alpha . \frac{k-1}{2} . \frac{m}{n}\right) \qquad \textbf{(B.6)}$$

$$r = rem\left(\left(i - \frac{k+1}{2}\right) . \frac{m}{n} + \frac{1}{2}\right) \qquad \textbf{(B.7)}$$

$$j_c = int\left(\left(i - \frac{k+1}{2}\right) . \frac{m}{n} + \frac{1}{2}\right) \qquad \textbf{(B.8)}$$

and

$$\delta = \begin{cases} 1 & h = 0 \\ 0 & otherwise \end{cases} \qquad \textbf{(B.8)}$$

In equations above ***int(.)*** denotes the largest integer whose magnitude does not exceed that of ***(.)*** , ***rem(.)*** is the difference between ***(.)*** and ***int(.)*** . $\alpha \in [0.0 , 1.0]$ is a constant that determines the number of location parameters to be taken in to account in calculating the average. The larger the value of α , the more parameters are used for computing the average. When $\alpha = 1.0$, the i^{th} knots are an average of $(k-1)m/n+1$ parameters. When $\alpha=0.0$ or h=0 the equations above simplly return linear interpolation results as $X_i = \mu_i$, $i=k+1, k+2, ...,n$. Throughout the present implementation $\alpha = 1$ is assumed.

Appendix C

Simulated Binary Crossover (SBX)

Simulated binary crossover (SBX) is a crossover designed for 'Real coded GAs' [19]. It enjoys much more flexibility in terms of search power as compared to the other crossovers reported in literature for real coded GAs. The most striking feature of this type of crossover is that it simulates the effect of standard binary crossover in a real-coded variables' domain.

Spread factor : In order to define the spread of the children points with respect to that of its parents, the term *spread factor* is used. It may be defined as the ratio of the spread of the children points to that of the parent points.

$$\beta = \left| \frac{c_1 - c_2}{p_1 - p_2} \right| \qquad \textbf{(C.1)}$$

SBX algorithm : Algorithm for calculating children points with the SBX crossover operator is given below

Input : p_1 , p_2 (Parent points)

Output : c_1 , c_2 (Children points)

- Generate a random number u such that $0 \leq u \leq 1$
- Calculate the spread factor,

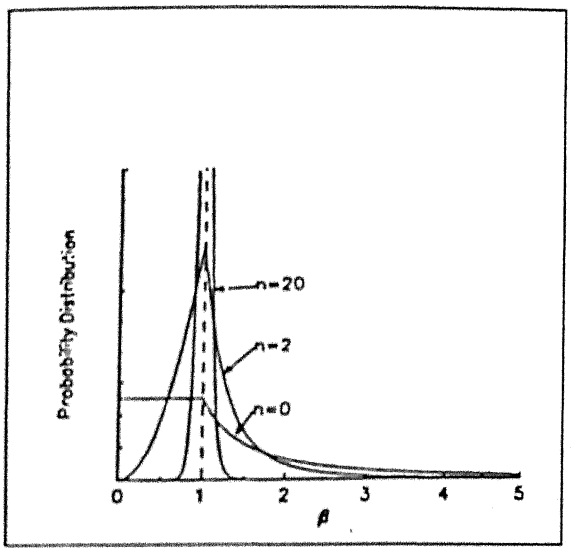
$$\beta(u) = \begin{cases} (2u)^{\frac{1}{(n+1)}} & \text{if } u \leq 0.5, \\ \left(\frac{1}{2(1-u)}\right)^{\frac{1}{(n+1)}} & \text{otherwise} \end{cases}$$

- Calculate children points ,

The number u used in the above algorithm represents the area under the curve from $\beta = 0$

$$c_1 = 0.5(p_1 + p_2) - 0.5\beta(u)|p_2 - p_1|$$

$$c_2 = 0.5(p_1 + p_2) + 0.5\beta(u)|p_2 - p_1|$$



FigureC.1 Probability Distribution for SBX

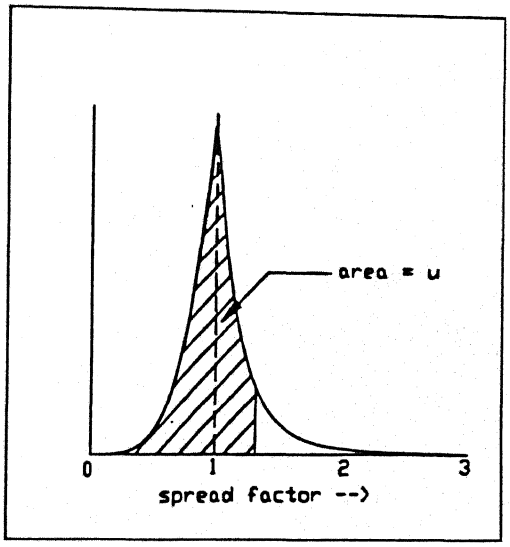


Figure C.2 Calculation of β from a Given Number u

to $\beta = \beta(u)$. The term n is known as distribution index. Figure C.1 shows the probability istributions of the crossover for various values of n.

Appendix D

Simplex Search Method

Simplex search method is a population based zero order search method proposed by Nelder and Mead. For a function of N variables, the method typically starts with a set of $N+1$ points. This set of $N+1$ points is called as the simplex. At each iteration, a new simplex is created from the current simplex by fixed transition rules. These transition rules are nothing but a set of simple operators such as reflection, contraction and expansion (Figure D.1). The objective function is evaluated at each of the $N+1$ points of the simplex. The points are ranked as the best, worst and the next to worst point. The worst point is projected a suitable

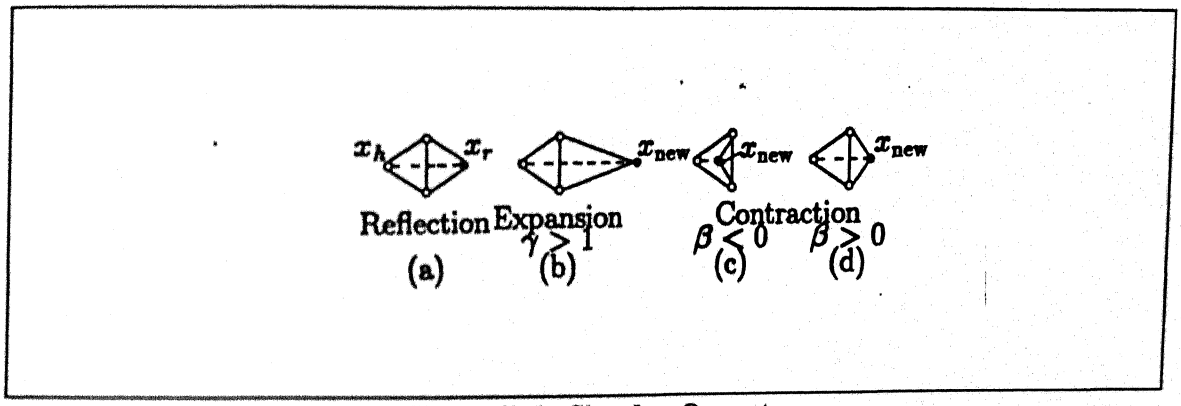


Figure D.1 Simplex Operators

distance through the centroid of the remaining points [17].

The algorithm of the method is listed below

Step 1 Choose $\gamma>1$, $\beta \in (0,1)$, and a termination parameter δ. Create an initial simplex using a base point and generating points around it using some scale factor.

Step 2 Find x_h (the worst point), x_l (the best point), and x_g (next to the worst point). Calculate

$$x_c = \frac{1}{N} \sum_{i=1, i \neq h}^{N+1} x_i$$

Step 3 Calculate the reflected point $x_r = 2x_c - x_h$. Set $x_{new} = x_r$.

If $f(x_r) < f(x_l)$, set $x_{new} = (1+\gamma)x_c - \gamma x_h$ (expansion).

Else if $f(x_r) >= f(x_h)$, set $x_{new} = (1-\beta)x_c + \beta x_h$ (contraction).

Else if $f(x_g) < f(x_r) < f(x_h)$, set $x_{new} = (1+\beta)x_c - \beta x_h$ (contraction).

Calculate $f(x_{new})$ and replace x_h by x_{new}.

Step 4 If

$$\left\{ \sum_{i=1}^{N+1} \frac{(f(x_i) - f(x_c))^2}{N+1} \right\}^{\frac{1}{2}} \leq \delta$$

terminate

else go to step 2.

ME-1996-M-ABB-OPT